Hierarchical N-mixture Models for species interactions: Empirical Data

Ben Weinstein - Stony Brook University

```
## [1] "Run Completed at 2016-08-29 15:45:42"
```

```
#reload if needed
#load("Observed.Rdata")
```

1 Load in data

```
#read in flower morphology data, comes from Nectar.R
fl.morph<-read.csv("InputData/FlowerMorphology.csv")

#use effective corolla where possible.
fl.morph$Corolla<-fl.morph$EffectiveCorolla

fl.morph[is.na(fl.morph$Corolla),"Corolla"]<-fl.morph[is.na(fl.morph$Corolla),"TotalCorolla"]

#First row is empty
fl.morph<-fl.morph[-1,]

#Bring in Hummingbird Morphology Dataset, comes from
hum.morph<-read.csv("InputData/HummingbirdMorphology.csv")

#taxonomy change, we are calling them Crowned Woodnymph's now.
hum.morph$English<-as.character(hum.morph$English)
hum.morph$English[hum.morph$English %in% "Green-crowned Woodnymph"]<-"Crowned Woodnymph"

#Bring in Interaction Matrix
int<-read.csv("InputData/HummingbirdInteractions.csv")

int$timestamp<-as.POSIXct(paste(int$Time,int$DateP),format="%H:%M:%S %Y-%m-%d")

#one date error
int[int$DateP %in% '2013-07-25',"Month"]<-7

#one duplicate camera error, perhaps two GPS records.
int<-int[!(int$ID %in% "FH1108" & int$Date_F %in% '2014-12-01'),]

#Correct known taxonomic disagreements, atleast compared to traits
int[int$Iplant_Double=="Alloplectus purpureus","Iplant_Double"]<-"Glossoloma purpureum"
int[int$Iplant_Double=="Capanea affinis","Iplant_Double"]<-"Kohleria affinis"
int[int$Iplant_Double=="Columnea cinerea","Iplant_Double"]<-"Columnea mastersonii"
int[int$Iplant_Double=="Alloplectus teuscheri","Iplant_Double"]<-"Drymonia teuscheri"
int[int$Iplant_Double=="Drymonia collegarum","Iplant_Double"]<-"Alloplectus tetragonoides"

#Some reasonable level of presences, 10 points
keep<-names(which(table(int$Hummingbird) > 10))

int<-int[int$Hummingbird %in% keep & !int$Hummingbird %in% c("Sparkling Violetear"),]

m.dat<-droplevels(int[colnames(int) %in%
c("ID","Video","Time","timestamp","Hummingbird","Sex","TransectID","Transect_R","Iplant_Double","P
erce","DateP","Month","ele","Type")])

#Does the data come from camera or transect?
m.dat$Type<-(is.na(m.dat$TransectID))*1

m.dat$Year<-years(as.Date(m.dat$DateP))
#one missing date
m.dat$Year[m.dat$Year %in% 2012]<-2013
m.dat$Year[m.dat$Year %in% 2106]<-2016
```

```
#Number of bird species
h_species<-nlevels(m.dat$Hummingbird)

#Number of plant species
plant_species<-nlevels(m.dat$Iplant_Double)

#Get english name
dath<-merge(m.dat,hum.morph, by.x="Hummingbird",by.y="English",keep=all)

#Merge to flowers
int.FLlevels<-levels(factor(dath$Iplant_Double))

#Which flowers are we missing info for?
missingTraits<-int.FLlevels[!int.FLlevels %in% fl.morph$X]

#print(paste("Missing Trait Information:",missingTraits))
dath<-merge(dath,fl.morph, by.x="Iplant_Double",by.y="X")

#Drop piercing events, since they don't represent correlation
#dath<-dath[!dath$Pierce %in% c("y","Y"),]
```

1.1 Match Species to Morphology

```
#observed traitmatching
traitmatchF<-abs(t(sapply(hum.morph$Bill,function(x){x-fl.morph$Corolla})))/10
rownames(traitmatchF)<-hum.morph$English
colnames(traitmatchF)<-fl.morph$Group.1
```

```
#match names #Round to 2 decimals #Convert to cm for winbugs, avoids numerical underflow
traitmatchT<-round(traitmatchF[rownames(traitmatchF) %in% dath$Hummingbird,colnames(traitmatchF)
 %in% dath$Iplant_Double],2)
traitmatchT<-traitmatchT[sort(rownames(traitmatchT)),sort(colnames(traitmatchT))]
```

1.2 Elevation ranges

Create a binary variable whether each observation was in a low elevation or high elevation transect. We have some species that just occur at the top of the gradient, and are not present in the sampling window of flowers at the low elevation.

Accounting for non-availability. We have to figure out which plants were sampled in which periods, and if it was sampled, the non-detection are 0 if it wasn't the non-detection are NA. then remove all the Na's.

```
elevH<-read.csv("InputData/HummingbirdElevation.csv",row.names=1)
colnames(elevH)[5]<-"Elevation"
elevH$Bird<-1:nrow(elevH)

#high elevation or low elevation
elevP<-read.csv("InputData/PlantElevation.csv",row.names=1)
colnames(elevP)[5]<-"Elevation"
elevP$Plant<-1:nrow(elevP)
elevP$Iplant_Double<-as.character(elevP$Iplant_Double)

#Merge to observed Data
#plants
dathp<-merge(dath,elevP,by="Iplant_Double")

#birds
datph<-merge(dathp,elevH,by="Hummingbird")
```

What elevation transect is each observation in? The camera data need to be inferred from the GPS point.

```
#cut working best on data.frame
datph<-as.data.frame(datph)

#which elevation bin is each observation within
labs<-paste(seq(1300,2500,200),seq(1500,2700,200),sep="_")

#for the couple points that have 1290 elevation, round up to 300 for convienance
datph$ele[datph$ele < 1300]<-1301

#make sure transect is a character
datph$Transect_R<-as.character(datph$Transect_R)
datph$Transect_R[is.na(datph$Transect_R)]<-
as.character(cut(datph[is.na(datph$Transect_R),]$ele,seq(1300,2700,200),labels=labs))

#Elev for the transects is the midpoint
tran_elev<-datph[datph$Survey_Type=='Transect',"Transect_R"]
datph[datph$Survey_Type=='Transect',"ele"]<-sapply(tran_elev,function(x){
  mean(as.numeric(str_split(x,"_")[[1]]))
})
```

1.2.1 Define Time Events

```
#ID for NA is holger transects, make the id's 1:n for each day of transect at each elevation, as
suming no elevation was split across days.
datph$ID<-as.character(datph$ID)
noid<-datph[is.na(datph$ID),]

id_topaste<-paste(noid$Month,noid$Year,"Transect",sep="_")
datph[which(is.na(datph$ID)),"ID"]<-id_topaste

#Create year month combination
datph$Time<-paste(datph$Month,datph$Year,sep="_")

#Label survey type
datph$Survey_Type<-NA

mt<-!is.na(datph$TransectID)*1
datph$Survey_Type[mt==1]<-"Transect"
datph$Survey_Type[!datph$Survey_Type %in% "Transect"]<-"Camera"
datph<-datph[datph$Survey_Type=="Camera",]

#time filter

#sort by timestamp
datph<-datph[order(datph$timestamp),]

dotime<-function(d){
  d$Timediff<-NA
  if(nrow(d)>1){
  for (x in 2:nrow(d)){
  d$Timediff[x]<-difftime(d$timestamp[x],d$timestamp[x-1],units="mins")
  }
  }
  return(d)
}

datph<-datph %>% group_by(ID,Hummingbird) %>% do(dotime(.))

#eliminate interaction by the same species within five minutes
paste("Removing ",length(which(datph$Timediff<5))," observations, not enough time since last obs
ervation of the same species",sep="")
```

```
## [1] "Removing 17 observations, not enough time since last observation of the same species"
```

```
datph<-datph[!1:nrow(datph) %in% which(datph$Timediff<5),]

#Day level
#add day ID
sdat<-split(datph,list(datph$ID),drop = T)

sdat<-lapply(sdat,function(x){
  x<-droplevels(x)
  x$Day<-as.numeric(as.factor(x$DateP))
  return(x)
})

indatraw<-rbind_all(sdat)

#Species names
for (x in 1:nrow(indatraw)){
  indatraw$Hummingbird[x]<-as.character(elevH[elevH$Bird %in% indatraw$Bird[x],"Hummingbird"])
  indatraw$Iplant_Double[x]<-as.character(elevP[elevP$Plant %in% indatraw$Plant[x],"Iplant_Doubl
e"])
}

#match the traits
traitmelt<-melt(traitmatchT)
colnames(traitmelt)<-c("Hummingbird","Iplant_Double","Traitmatch")

#dummy presence variable
indatraw$Yobs<-1

#prune columsn to make more readable
indatraw<-indatraw[,c("Hummingbird","Iplant_Double","ID","Time","Month","Year","Transect_R","el
e","DateP","Yobs","Day","Survey_Type","Pierce")]
```

1.3 Summarize daily interactions

To estimate the daily detectability, there can only be a max of one interaction per day. We use mean elevation to average across observations within a transect

```
indatraw<-indatraw %>% group_by(Hummingbird,Iplant_Double,ID,Day) %>% summarize(Yobs=sum(Yobs),T
ime=unique(Time),Transect_R=unique(Transect_R),Month=unique(Month),Year=unique(Year),ele=mean(el
e),DateP=unique(DateP),Survey_Type=unique(Survey_Type)) %>% ungroup()
```

1.4 Absences - accounting for non-detection

We have more information than just the presences, given species elevation ranges, we have absences as well. Absences are birds that occur at the elevation of the plant sample, but were not recorded feeding on the flower.

```
#Only non-detections are real 0's, the rest are NA's and are removed.
#Plants not surveyed in that time period
#Hummingbirds not present at that elevation

#For each ID
Time<-unique(indatraw$Time)

#absences data frame
absences<-list()

for(t in Time){
  IDlist<-unlist(unique(indatraw[indatraw$Time ==t,"ID"]))

  for (j in IDlist){
  #Which plants were sampled
  a<-indatraw %>% filter(Time==t,ID==j)

  #For each sampled transect
  trans<-unique(a$Transect_R)

  if(!length(trans)==0){
    for(transect in trans){

    #for each date
    datec<-a %>% filter(Transect_R %in% transect)
    datecam<-unique(datec$DateP)
    }} else{
      datecam<-a %>% distinct(DateP) %>% .$DateP
    }
    for(Date in datecam){

    #for each plant along that transect at that date
    pres<-a %>% filter(DateP %in% Date) %>% distinct(Iplant_Double) %>% .$Iplant_Double

    #Which day in sampling
    dday<-a %>% filter(Transect_R %in% transect,DateP %in% Date) %>% distinct(Day) %>% .$Day

      for (plant in pres){
        #Get mean elevation of that plant record
        camelev<- a %>% filter(Transect_R %in% transect,DateP %in% Date,Iplant_Double %in%
plant) %>% .$ele %>% mean()

        #Which birds are present at that observation
        predh<-elevH[((elevH$Low < camelev) & (camelev < elevH$High)),"Hummingbird"]

        #remove the ones seen on that plant
        hum_present<-a %>% filter(Transect_R %in% transect,DateP %in% Date,Iplant_Double %in% pl
ant) %>% .$Hummingbird
        abbh<-predh[!predh %in% hum_present]
        if(length(abbh)==0){next}

        #Make absences from those )(cat not the best)
        add_absences<-data.frame(Hummingbird=abbh,Iplant_Double=plant,Time=t,ID=j,DateP=Date,Mon
```

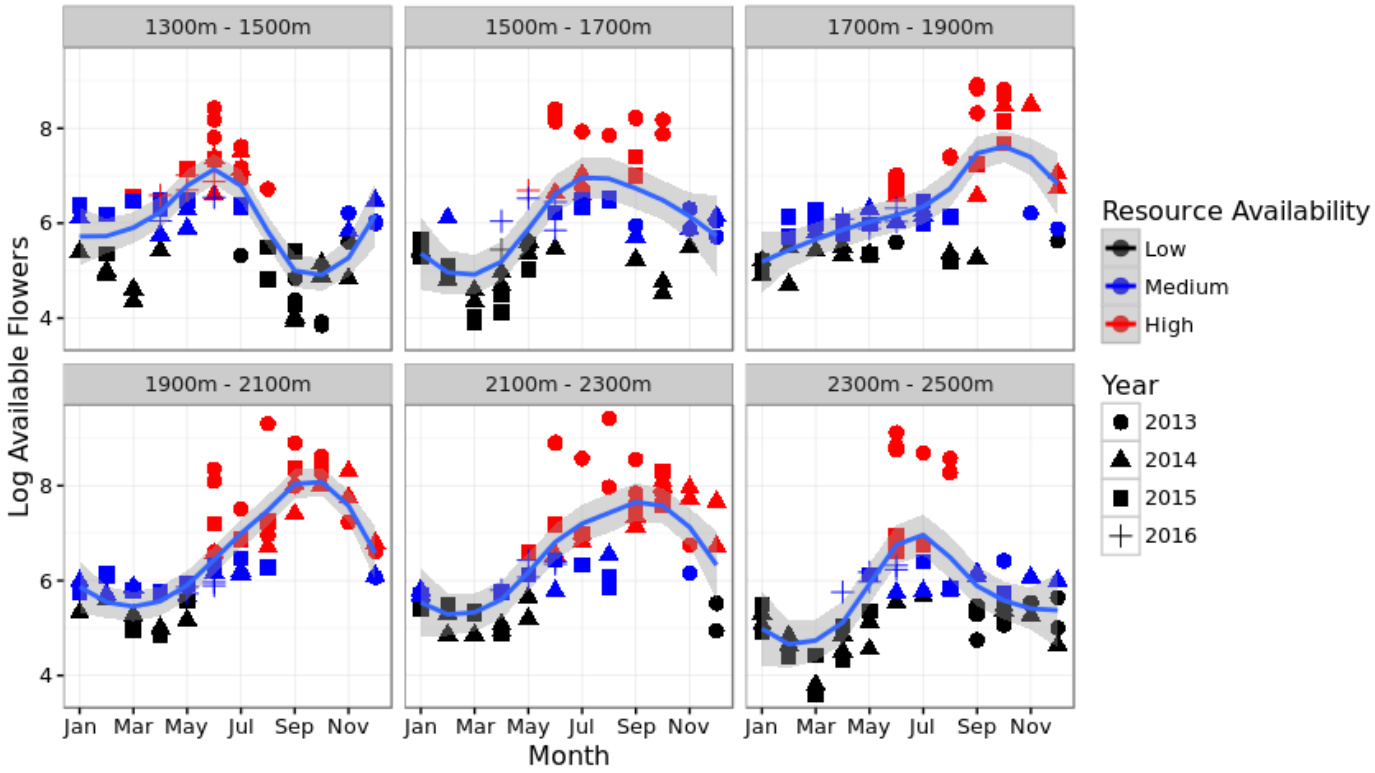
```
th=min(a$Month),Year=unique(a$Year),Transect_R=transect,ele=camelev,Day=unique(dday),Survey_Type=u
ique(a$Survey_Type),Yobs=0)
        absences<-append(absences,list(add_absences))
      }
    }
  }
}

indatab<-rbind_all(absences)

#merge with original data
indat<-rbind_all(list(indatraw,indatab))
```

```
#Get trait information
#match the traits
indat<-merge(indat,traitmelt,by=c("Hummingbird","Iplant_Double"))
```

2 Resources at each month

In our model the covariate is indexed at the scale at which the latent count is considered fixed. This means we need the resource availability per month across the entire elevation gradient for each point.

```
#Get flower transect data
full.fl<-read.csv("InputData/FlowerTransectClean.csv")[,-1]

 #month should be capital
colnames(full.fl)[colnames(full.fl) %in% "month"]<-"Month"

#group by month and replicate, remove date errors by making a max of 10 flowers, couple times wh
ere the gps places it in wrong transect by 1 to 2 meters.
flower.month<-group_by(full.fl,Month,Year,Transect_R,Date_F) %>% dplyr::summarise(Flowers=sum(To
tal_Flowers,na.rm=TRUE))  %>% filter(Flowers>20)

#Make month abbreviation column, with the right order
flower.month$Month.a<-factor(month.abb[flower.month$Month],month.abb[c(1:12)])

#Make year factor column
flower.month$Year<-as.factor(flower.month$Year)

#get quantile for each transect
#thresh<-melt(group_by(flower.month) %>% summarize(Threshold=quantile(Flowers,0.5)))
flower.month$R<-cut(flower.month$Flowers,breaks=c(0,quantile(flower.month$Flowers,0.33),quantile(f
ower.month$Flowers,0.66),max(flower.month$Flowers)),label=c("Low","Medium","High"))

#fix the levels
flower.month$PTransect_R<-flower.month$Transect_R
levels(flower.month$PTransect_R)<-c("1300m - 1500m", "1500m - 1700m","1700m - 1900m","1900m - 21
00m","2100m - 2300m","2300m - 2500m")
#plot

ggplot(flower.month,aes(x=Month.a,log(Flowers),col=R,shape=as.factor(Year))) +
geom_point(size=3) + theme_bw()  + geom_smooth(aes(group=1)) + ylab("Flowers") + xlab("Month") +
 facet_wrap(~PTransect_R) + labs(shape="Year", y= "Log Available Flowers") + scale_x_discrete(br
eaks=month.abb[seq(1,12,2)]) + scale_color_manual(labels=c("Low","Medium","High"),values=c("blac
k","blue","red")) + labs(col="Resource Availability")
```

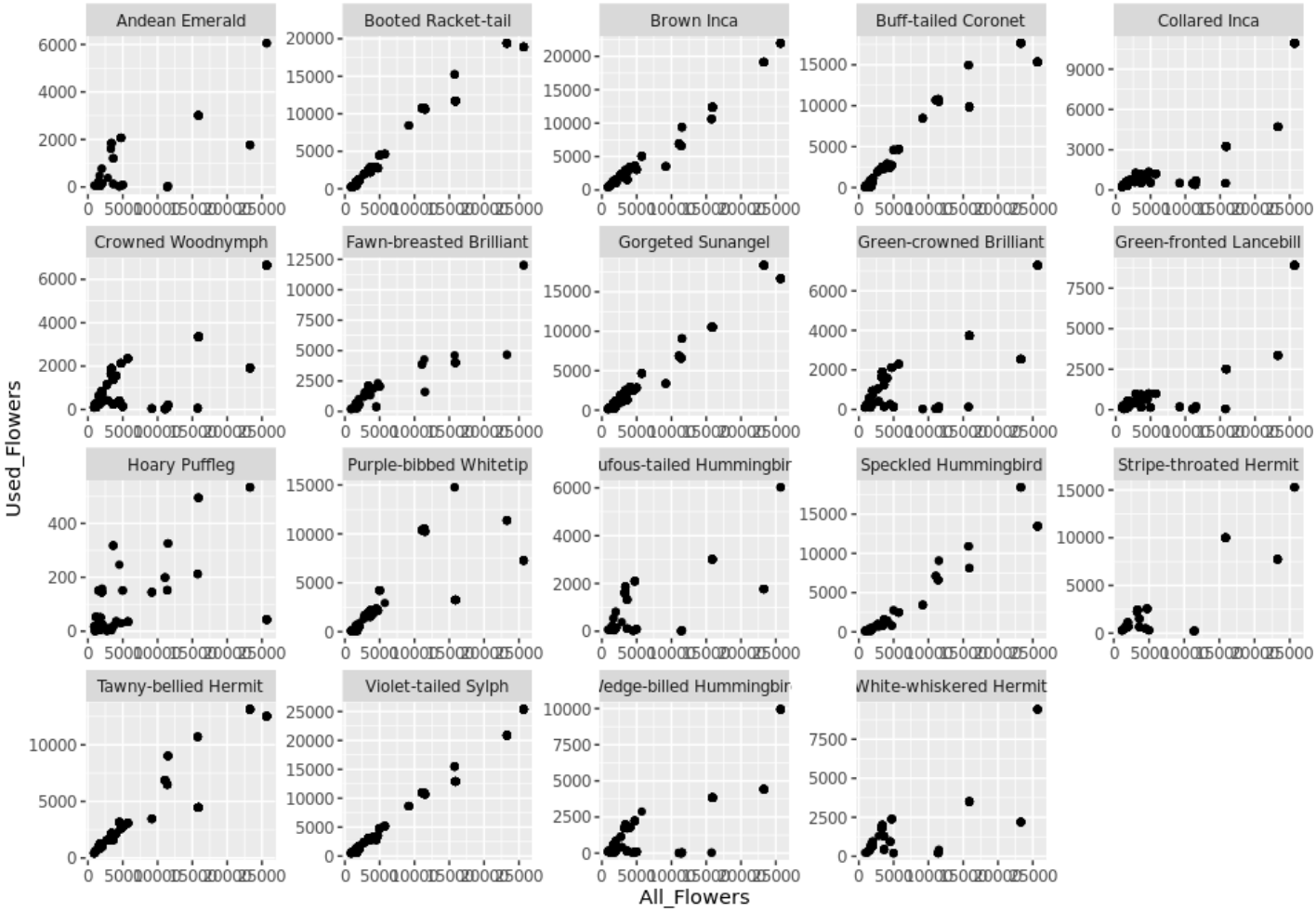



```
ggsave("Figures/FlowerMonth.jpeg",dpi=600,height=5,width=9)

#turn min and max elvation into seperate columns for the range
flower.month$minElev<-as.numeric(str_extract(flower.month$Transect_R,"(\\d+)"))
flower.month$maxElev<-as.numeric(str_match(flower.month$Transect_R,"(\\d+)_(\\d+)")[,3])
```

```
indat$All_Flowers<-NA
indat$Used_Flowers<-NA
indat$FlowerA<-NA

#Resource list for each species.
slist<-int %>% group_by(Hummingbird,Iplant_Double) %>% distinct() %>%
dplyr::select(Hummingbird,Iplant_Double) %>% arrange(Hummingbird)

#Create time ID for flower transects
full.fl$Time<-paste(full.fl$Month,full.fl$Year,sep="_")

#all flowers for each ID period
allF<-full.fl %>% group_by(Month,Year,Transect_R,Date_F) %>% summarize(n=sum(Total_Flowers,na.rm=T
)) %>% summarize(mn=mean(n)) %>% summarize(F=sum(mn)) %>% as.data.frame()

#Individual flowers for each ID period
indF<-full.fl %>% group_by(Iplant_Double,Month,Year,Transect_R,Date_F) %>% summarize(n=sum(Total
_Flowers,na.rm=T)) %>% summarize(mn=mean(n)) %>% summarize(F=sum(mn)) %>% as.data.frame()

for (x in 1:nrow(indat)){

#All flowers
 indat$All_Flowers[x]<-allF[allF$Month %in% indat$Month[x] & allF$Year %in% indat$Year[x],"F"]

 #filter by species used by hummingbird
 sp_list<-slist[slist$Hummingbird %in% indat$Hummingbird[x],"Iplant_Double"]

 indat$Used_Flowers[x]<-sum(indF[indF$Iplant_Double %in% sp_list$Iplant_Double & indF$Month %in%
 indat$Month[x] & indF$Year %in% indat$Year[x],"F"])

  #just the abundance of that species
  indat$FlowerA[x]<-sum(indF[indF$Iplant_Double %in% indat$Iplant_Double[x] & indF$Month %in% in
dat$Month[x] & indF$Year %in% indat$Year[x],"F"])

}
```

2.0.1 Relationship between resource measures

```
ggplot(indat,aes(x=All_Flowers,y=Used_Flowers)) + geom_point() +
facet_wrap(~Hummingbird,scales="free")
```



2.1 Binary Measures of Resources

```
#All Resources
#indat$BAll_Flowers<-(indat$Month  %in% c("6","7","8","9","10"))*1

indat$BAll_Flowers<-(indat$All_Flowers > quantile(indat$All_Flowers,0.5))*1

qthresh<-indat %>% group_by(Hummingbird) %>% summarize(UThresh=quantile(Used_Flowers,0.75))

indat<-merge(indat,qthresh)
indat$BUsed_Flowers<-(indat$Used_Flowers > indat$UThresh)*1

fthresh<-indat %>% group_by(Hummingbird) %>% summarize(FThresh=mean(FlowerA))
indat<-merge(indat,fthresh)
indat$BFlowerA<-(indat$FlowerA > indat$FThresh)*1

#merge with flower month, split by elevation, mean per month
sflowers<-flower.month %>% group_by(Transect_R,Month,Year) %>% summarize(Flowers=mean(Flowers))
sflowers$R<-cut(sflowers$Flowers,breaks=c(0,quantile(sflowers$Flowers,0.33),quantile(sflowers$Fl
owers,0.66),max(sflowers$Flowers)),label=c("Low","Medium","High"))

indat<-merge(indat,sflowers,c("Transect_R","Month","Year"))
```

```
#Combine resources with observed data
f<-(indat$Survey_Type=="Camera")*1
f[f==0]<-NA
indat$Camera<-indat$Yobs * f

f<-(indat$Survey_Type=="Transect")*1
f[f==0]<-NA
indat$Transect<-indat$Yobs * f
```

Reformat index for jags. Jags needs a vector of input species 1:n with no breaks.

```
#Subset if model testing here
#indat<-indat[indat$Hummingbird %in% c("White-whiskered Hermit","Brown Inca"),]

#Easiest to work with jags as numeric ordinal values
indat$Hummingbird<-as.factor(indat$Hummingbird)
indat$Iplant_Double<-as.factor(indat$Iplant_Double)
indat$jBird<-as.numeric(indat$Hummingbird)
indat$jPlant<-as.numeric(indat$Iplant_Double)

jagsIndexBird<-data.frame(Hummingbird=levels(indat$Hummingbird),jBird=1:length(levels(indat$Humm
ingbird)))
jagsIndexPlants<-data.frame(Iplant_Double=levels(indat$Iplant_Double),jPlant=1:length(levels(ind
at$Iplant_Double)))

#Similiarly, the trait matrix needs to reflect this indexing.
jTraitmatch<-traitmatchT[rownames(traitmatchT) %in% unique(indat$Hummingbird),colnames(traitmatc
hT) %in% unique(indat$Iplant_Double)]
```

```
indat<-droplevels(indat)

#Turn Time and ID into numeric indexes
indat$jTime<-as.numeric(as.factor(indat$Time))
indat$jID<-as.numeric(as.factor(indat$ID))

#index resources
indat$scaledR<-as.numeric(indat$FlowerA)/100
resourcemat<-indat %>% group_by(jBird,jPlant,jID) %>% summarize(v=max(scaledR))  %>% acast(jBird
 ~ jPlant ~ jID,value.var='v',fill=0)

#index position
indat$Index<-1:nrow(indat)
```

3 Hierarcichal N-mixture Model

3.1 Traits

For hummingbird i visiting plant j recorded by camera k on day d:

$$Y_{i,j,k,d} \sim Binom(N_{i,j,k}, \omega_i)$$

$$N_{i,j,k} \sim Pois(\lambda_{i,j,k})$$

$$log(\lambda_{i,j}) < -\alpha_i + \beta_{1,i} * |Bill_i - Corolla_j|$$

3.2 Abundance

Replace:

$$log(\lambda_{i,j}) < -\alpha_i + \beta_{1,i} * |Bill_i - Corolla_j|$$

with:

$$log(\lambda_{i,j}) < -\alpha_i + \beta_{1,i} * Resources_{i,j,k}$$

Priors

Please recall that jags parameterizes models using precision, not sd (precision = 1/sd^2)

$$\omega_i \sim (\mu_\omega, \tau_\omega)$$

$$\mu_\omega \sim Normal(0, 0.5)$$

_{} Uniform(0,10)

$$\alpha_i \sim Normal(\mu_\alpha, \tau_\alpha)$$

$$\beta_{1,i} \sim Normal(\mu_{\beta_1}, \tau_{\beta_1})$$

$$\beta_{2,i} \sim Normal(\mu_{\beta_2}, \tau_{\beta_2})$$

Hyperpriors

$$\mu_\alpha \sim Normal(0, 0.0001)$$

$$\mu_{\beta_1} \sim Normal(0, 0.0001)$$

$$\tau_\alpha \sim Half-T(0.0001, 0.0001)$$

$$\tau_{\beta_1} = \sqrt[2]{\frac{1}{\sigma_{\beta_1}}}$$

$$\tau_{\beta_2} = \sqrt[2]{\frac{1}{\sigma_{\beta_2}}}$$

$$\sigma_\alpha = \sqrt[2]{\frac{1}{\tau_\alpha}}$$

$$\sigma_{\beta_1} \sim Half-T(0, 1)$$

$$\sigma_{\beta_2} \sim Half-T(0, 1)$$

4 Poisson GLMM

```
runs<-50000

#Source model
source("Bayesian/NoDetectNmixturePoissonRagged.R")

#print model
print.noquote(readLines("Bayesian//NoDetectNmixturePoissonRagged.R"))
```

```
##  [1]
##  [2] sink("Bayesian/NoDetectNmixturePoissonRagged.jags")
##  [3]
##  [4] cat("
##  [5]     model {
##  [6]     #Compute intensity for each pair of birds and plants
##  [7]     for (i in 1:Birds){
##  [8]     for (j in 1:Plants){
##  [9]     for (k in 1:Cameras){
## [10]
## [11]     #Process Model with log normal overdispersion
## [12]     log(lambda[i,j,k])<-alpha[i] + beta1[i] * Traitmatch[i,j] + epsilon[i,j,k]
## [13]
## [14]     #Log transformed variance in counts
## [15]     epsilon[i,j,k] ~ dnorm(0,tauE)
## [16]
## [17]     }
## [18]     }
## [19]     }
## [20]
## [21]     for (x in 1:Nobs){
## [22]
## [23]       # Observed State
## [24]       Yobs[x] ~ dpois(lambda[Bird[x],Plant[x],Camera[x]])
## [25]
## [26]       #Assess Model Fit
## [27]
## [28]       #Fit discrepancy statistics
## [29]       eval[x]<-lambda[Bird[x],Plant[x],Camera[x]]
## [30]       E[x]<-pow((Yobs[x]-eval[x]),2)/(eval[x]+0.5)
## [31]
## [32]       ynew[x]~dpois(lambda[Bird[x],Plant[x],Camera[x]])
## [33]       E.new[x]<-pow((ynew[x]-eval[x]),2)/(eval[x]+0.5)
## [34]
## [35]       }
## [36]
## [37]     #Process Model
## [38]     #Species level priors
## [39]     for (i in 1:Birds){
## [40]
## [41]     #Intercept
## [42]     alpha[i] ~ dnorm(alpha_mu,alpha_tau)
## [43]
## [44]     #Traits slope
## [45]     beta1[i] ~ dnorm(beta1_mu,beta1_tau)
## [46]     }
## [47]
## [48]     #Group process priors
## [49]
## [50]     #Intercept
## [51]     alpha_mu ~ dnorm(0,0.001)
## [52]     alpha_tau ~ dt(0,1,1)I(0,)
## [53]     alpha_sigma<-pow(1/alpha_tau,0.5)
```

```
## [54]
## [55]     #Trait
## [56]     beta1_mu~dnorm(0,0.001)
## [57]     beta1_tau ~ dt(0,1,1)I(0,)
## [58]     beta1_sigma<-pow(1/beta1_tau,0.5)
## [59]
## [60]     #Overdispersion
## [61]     #Overdispersion
## [62]     tauSigma ~ dunif(0,0.5)
## [63]     tauE <- pow(1/tauSigma,2)
## [64]
## [65]     #derived posterior check
## [66]     fit<-sum(E[]) #Discrepancy for the observed data
## [67]     fitnew<-sum(E.new[])
## [68]
## [69]
## [70]     }
## [71]     ",fill=TRUE)
## [72]
## [73] sink()
```

```
  #Data objects for parallel run
  Yobs=indat$Yobs
  Bird=indat$jBird
  Birds=max(indat$jBird)
  Plant=indat$jPlant
  Plants=max(indat$jPlant)
  Camera=indat$jID
  Cameras=max(indat$jID)
  Traitmatch=jTraitmatch
  Nobs=length(indat$Yobs)
  resources=resourcemat

  #A blank Y matrix - all present
  Ninit<-array(dim=c(Birds,Plants,Cameras),data=max(indat$Yobs)+1)

  #Inits
  InitStage <- function() {list(beta1=rep(0,Birds),alpha=rep(0,Birds),alpha_mu=0,beta1_mu=0,exp_
lambda=Ninit)}

  #Parameters to track
  ParsStage <- c("alpha","beta1","alpha_mu","beta1_mu","ynew","fit","fitnew","tauE")

  #MCMC options
  ni <- runs  # number of draws from the posterior
  nt <- 10   #thinning rate
  nb <- max(0,runs-3000) # number to discard for burn-in
  nc <- 2  # number of chains

  Dat<-list("Yobs","Bird","Plant","Plants","Camera","Cameras","Traitmatch","Birds","Ninit","Nob
s","nb","nt","nc","ni")

  system.time(m2_niave<-jags.parallel(Dat,InitStage,ParsStage,model.file="Bayesian/NoDetectNmixt
urePoissonRagged.jags", n.iter=ni,n.burnin=nb,n.chains=nc,n.thin=nt))
```

```
##     user   system  elapsed
##    5.097    0.207 1210.048
```

```
#recompile if needed
load.module("dic")
runs<-100000
recompile(m2_niave)
m2_niave<-update(m2_niave,n.iter=runs,n.burnin=runs*.9,n.thin = 5)
```

```
pars_dniave<-extract_par(m2_niave,data=indat,Bird="jBird",Plant="jPlant")
pars_dniave$Model<-"Poisson GLMM"
rm(m2_niave)
gc()
```

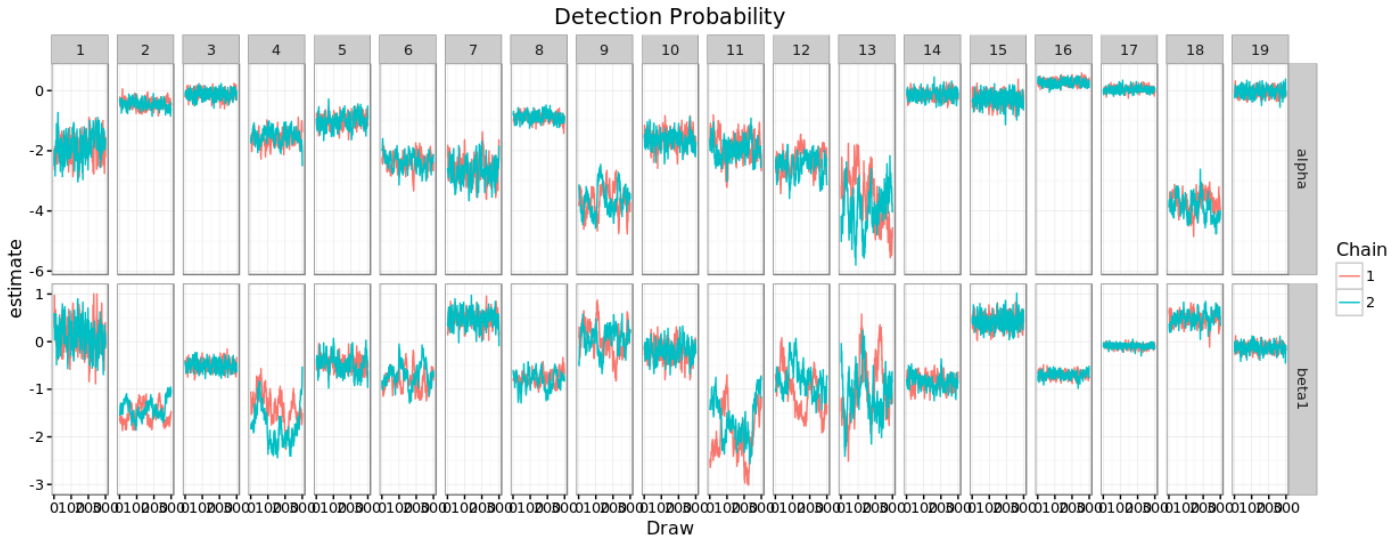
```
##            used  (Mb) gc trigger  (Mb)  max used  (Mb)
## Ncells  1677871  89.7    5295264 282.8   7433048 397.0
## Vcells 32811567 250.4   96467192 736.0 120174006 916.9
```

```
#write to file
#write.csv(pars_dniave,"OutData/GLMM.csv")
```

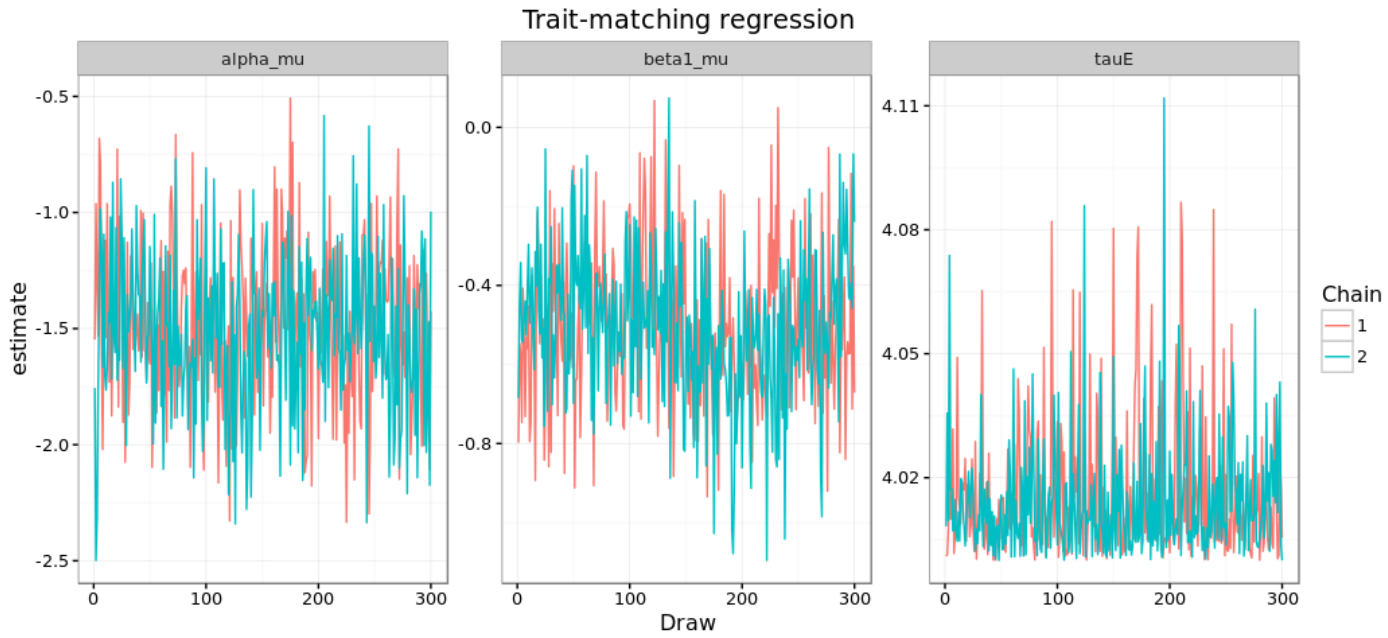
```
#read from file is needed
pars_dniave<-read.csv("OutData/GLMM.csv",row.names = 1)
```

4.1 Assess Convergence

```
###Chains
ggplot(pars_dniave[pars_dniave$par %in% c("alpha","beta1"),],aes(x=Draw,y=estimate,col=as.factor(C
ain))) + geom_line() + facet_grid(par~species,scale="free") + theme_bw() + labs(col="Chain") + g
gtitle("Detection Probability")
```




```
ggplot(pars_dniave[pars_dniave$par %in% c("beta1_mu","sigma_alpha","beta1_sigma","alpha_mu","tau
E"),],aes(x=Draw,y=estimate,col=as.factor(Chain))) + geom_line() + theme_bw() +
labs(col="Chain") + ggtitle("Trait-matching regression") + facet_wrap(~par,scales="free")
```



5 N-mixture

5.1 Traits

```
runs<-100000

#Source model
source("Bayesian/NmixturePoissonRagged.R")

#print model
print.noquote(readLines("Bayesian//NmixturePoissonRagged.R"))
```

```
##  [1]
##  [2] sink("Bayesian/NmixturePoissonRagged.jags")
##  [3]
##  [4] cat("
##  [5]     model {
##  [6]     #Compute intensity for each pair of birds and plants
##  [7]     for (i in 1:Birds){
##  [8]     for (j in 1:Plants){
##  [9]     for (k in 1:Times){
## [10]
## [11]     #Process Model with log normal overdispersion
## [12]     #mean intensity
## [13]
## [14]     #log transformed variance
## [15]     log(lambda[i,j,k]) <- alpha[i] + beta1[i] * Traitmatch[i,j] +  epsilon[i,j,k]
## [16]     epsilon[i,j,k] ~ dnorm(0,tauE)
## [17]
## [18]     #For each Time - there is a latent count, log transformed intensity
## [19]     N[i,j,k] ~ dpois(lambda[i,j,k])
## [20]     }
## [21]     }
## [22]     }
## [23]
## [24]     #Observed counts for each day of sampling at that Time
## [25]     for (x in 1:Nobs){
## [26]
## [27]     #Observation Process
## [28]     Yobs[x] ~ dbin(detect[Bird[x]],N[Bird[x],Plant[x],Time[x]])
## [29]
## [30]     #Assess Model Fit
## [31]
## [32]     #Fit discrepancy statistics
## [33]     eval[x]<-detect[Bird[x]]*N[Bird[x],Plant[x],Time[x]]
## [34]     E[x]<-pow((Yobs[x]-eval[x]),2)/(eval[x]+0.5)
## [35]
## [36]     ynew[x]~dbin(detect[Bird[x]],N[Bird[x],Plant[x],Time[x]])
## [37]     E.new[x]<-pow((ynew[x]-eval[x]),2)/(eval[x]+0.5)
## [38]
## [39]     }
## [40]
## [41]     ###Priors###
## [42]     #Observation model
## [43]     #Detect priors, logit transformed - Following lunn 2012 p85
## [44]
## [45]     for(x in 1:Birds){
## [46]     #For Cameras
## [47]     logit(detect[x])<-dcam[x]
## [48]     dcam[x]~dnorm(omega,omega_tau)
## [49]     }
## [50]
## [51]     #Process Model
## [52]     #Species level priors
## [53]     for (i in 1:Birds){
```

```
## [54]
## [55]     #Intercept
## [56]     alpha[i] ~ dnorm(alpha_mu,alpha_tau)
## [57]
## [58]     #Traits slope
## [59]     beta1[i] ~ dnorm(beta1_mu,beta1_tau)
## [60] }
## [61]
## [62]     #Group Detection Prior
## [63]     omega ~ dnorm(0,0.386)
## [64]     omega_tau ~ dt(0,1,1)I(0,)
## [65]
## [66]     #Group process priors
## [67]
## [68]     #Intercept
## [69]     alpha_mu ~ dnorm(0,0.01)
## [70]     alpha_tau ~ dt(0,1,1)I(0,)
## [71]     alpha_sigma<-pow(1/alpha_tau,0.5)
## [72]
## [73]     #Trait
## [74]     beta1_mu~dnorm(0,0.01)
## [75]     beta1_tau ~ dt(0,1,1)I(0,)
## [76]     beta1_sigma<-pow(1/beta1_tau,0.5)
## [77]
## [78]     #Overdispersion - can't go too low or log density will wander if into INF
## [79]     tauSigma ~ dunif(0,0.5)
## [80]     tauE <- pow(1/tauSigma,2)
## [81]
## [82]     #derived posterior check
## [83]     fit<-sum(E[]) #Discrepancy for the observed data
## [84]     fitnew<-sum(E.new[])
## [85]
## [86]
## [87]     }
## [88]     ",fill=TRUE)
## [89]
## [90] sink()
```

```
  #for parallel run
  Yobs=indat$Yobs
  Bird=indat$jBird
  Plant=indat$jPlant
  Time=indat$jID
  Times=max(indat$jID)
  Traitmatch=jTraitmatch
  Birds=max(indat$jBird)
  Plants=max(indat$jPlant)
  Nobs=length(indat$Yobs)

  #A blank Y matrix - all present
  Ninit<-array(dim=c(Birds,Plants,Times),data=max(indat$Yobs)+1)

  #Inits
  InitStage <- function() {list(beta1=rep(0,Birds),alpha=rep(0,Birds),N=Ninit,beta1_mu=0,exp_lam
bda=Ninit)}

  #Parameters to track
  ParsStage <- c("detect","alpha","beta1","alpha_mu","beta1_sigma","beta1_mu","ynew","fit","fitn
ew","E","tauE")

  #MCMC options
  ni <- runs  # number of draws from the posterior
  nt <- 10   #thinning rate
  nb <- max(0,runs-3000) # number to discard for burn-in
  nc <- 2  # number of chains

  Dat<-list("Yobs","Bird","Plant","Plants","Traitmatch","Birds","Nobs","Ninit","Time","Times","n
b","nc","ni","nt")

    system.time(traits<-jags.parallel(Dat,InitStage,ParsStage,model.file="Bayesian/NmixturePoiss
onRagged.jags",n.thin=nt, n.iter=ni,n.burnin=nb,n.chains=nc))
```

```
##     user   system  elapsed
##    8.594    0.168 3898.762
```

```
#recompile if needed
load.module("dic")
runs<-100000
recompile(traits)
traits<-update(traits,n.iter=runs,n.burnin=runs*.9,n.thin=5)
```

```
#extract par to data.frame
pars_detect_traits<-extract_par(traits,data=indat,Bird="jBird",Plant="jPlant")

rm(traits)
gc()
```

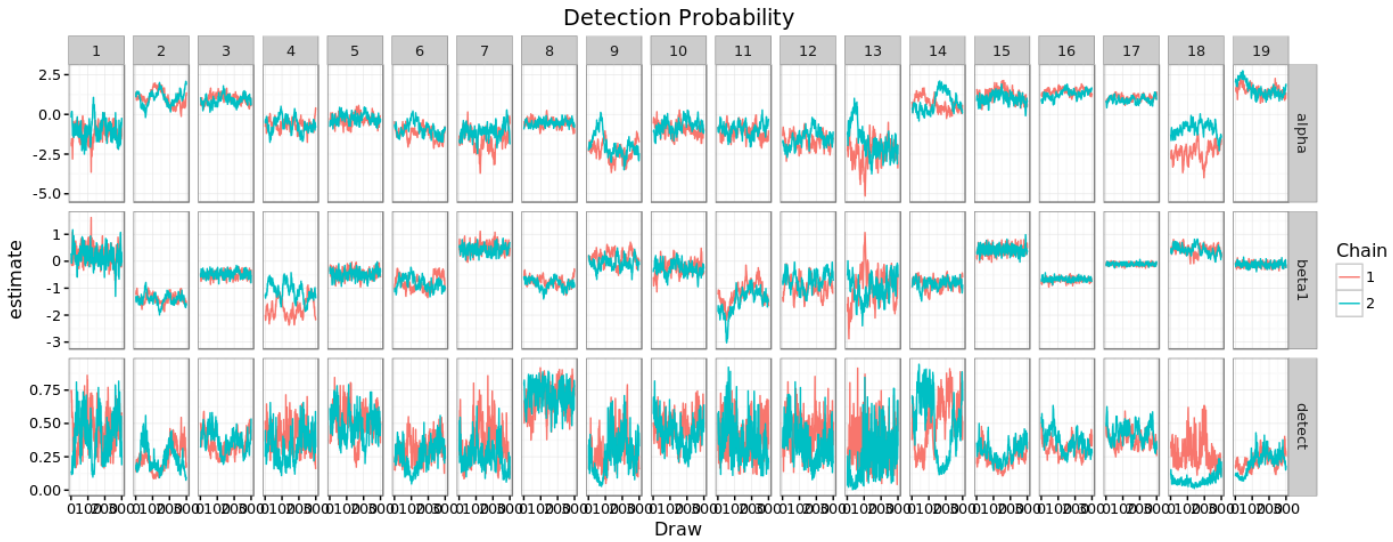
```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  1688115  90.2    8623340  460.6  13167343  703.3
## Vcells 79996684 610.4  240863060 1837.7 240366650 1833.9
```

```
#name
pars_detect_traits$Model<-"N-mixture: Traits"
#write.csv(pars_detect_traits,"OutData/Traits.csv")
```

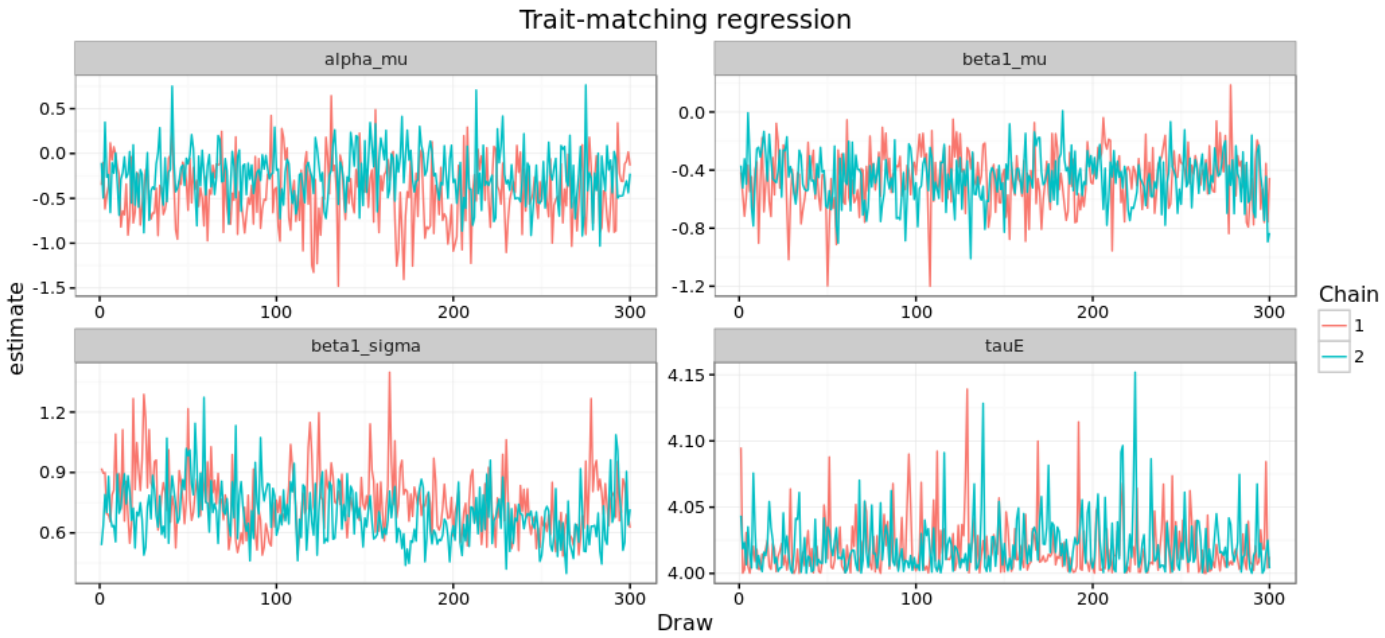
```
#read from file if needed
pars_detect_traits<-read.csv("OutData/Traits.csv",row.names=1)
```

5.1.1 Assess Convergence

```
###Chains
ggplot(pars_detect_traits[pars_detect_traits$par %in% c("detect","alpha","beta1"),],aes(x=Draw,y=e
timate,col=as.factor(Chain))) + geom_line() + facet_grid(par~species,scale="free") + theme_bw()
+ labs(col="Chain") + ggtitle("Detection Probability")
```




```
ggplot(pars_detect_traits[pars_detect_traits$par %in% c("beta1_mu","alpha_mu","sigma_alpha","bet
a1_sigma","tauE"),],aes(x=Draw,y=estimate,col=as.factor(Chain))) + geom_line() + theme_bw() + la
bs(col="Chain") + ggtitle("Trait-matching regression") + facet_wrap(~par,scales="free")
```



5.2 Abundance

```
runs<-100000

#Source model
source("Bayesian/NmixturePoissonRagged_Abundance.R")

#print model
print.noquote(readLines("Bayesian//NmixturePoissonRagged_Abundance.R"))
```

```
##  [1]
##  [2] sink("Bayesian/NmixturePoissonRagged_Abundance.jags")
##  [3]
##  [4] cat("
##  [5]     model {
##  [6]     #Compute intensity for each pair of birds and plants
##  [7]     for (i in 1:Birds){
##  [8]     for (j in 1:Plants){
##  [9]     for (k in 1:Times){
## [10]
## [11]     #Process Model with log normal overdispersion
## [12]     log(lambda[i,j,k])<-alpha[i] + beta1[i] * resources[i,j,k] + epsilon[i,j,k]
## [13]
## [14]     #variance
## [15]     epsilon[i,j,k] ~ dnorm(0,tauE)
## [16]
## [17]     #For each Time - there is a latent count, log transformed.
## [18]     N[i,j,k] ~ dpois(lambda[i,j,k])
## [19]     }
## [20]     }
## [21]     }
## [22]
## [23]
## [24]     #Observed counts for each day of sampling at that Time
## [25]     for (x in 1:Nobs){
## [26]
## [27]     #Observation Process
## [28]     Yobs[x] ~ dbin(detect[Bird[x]],N[Bird[x],Plant[x],Time[x]])
## [29]
## [30]     #Assess Model Fit
## [31]
## [32]     #Fit discrepancy statistics
## [33]     eval[x]<-detect[Bird[x]]*N[Bird[x],Plant[x],Time[x]]
## [34]     E[x]<-pow((Yobs[x]-eval[x]),2)/(eval[x]+0.5)
## [35]
## [36]     ynew[x]~dbin(detect[Bird[x]],N[Bird[x],Plant[x],Time[x]])
## [37]     E.new[x]<-pow((ynew[x]-eval[x]),2)/(eval[x]+0.5)
## [38]
## [39]     }
## [40]
## [41]     #Priors
## [42]     #Observation model
## [43]     #Detect priors, logit transformed - Following lunn 2012 p85
## [44]
## [45]     for(x in 1:Birds){
## [46]     #For Cameras
## [47]     logit(detect[x])<-dcam[x]
## [48]     dcam[x]~dnorm(omega,omega_tau)
## [49]     }
## [50]
## [51]
## [52]     #Process Model
## [53]     #Species level priors
```

```
## [54]     for (i in 1:Birds){
## [55]
## [56]     #Intercept
## [57]     alpha[i] ~ dnorm(alpha_mu,alpha_tau)
## [58]
## [59]     #Traits slope
## [60]     beta1[i] ~ dnorm(beta1_mu,beta1_tau)
## [61]     }
## [62]
## [63]     #Group Detection Prior
## [64]     omega ~ dnorm(0,0.386)
## [65]     omega_tau ~ dt(0,1,1)I(0,)
## [66]
## [67]     #Group process priors
## [68]
## [69]     #Intercept
## [70]     alpha_mu ~ dnorm(0,0.01)
## [71]     alpha_tau ~ dt(0,1,1)I(0,)
## [72]     alpha_sigma<-pow(1/alpha_tau,0.5)
## [73]
## [74]     #SLope
## [75]     beta1_mu~dnorm(0,0.01)
## [76]     beta1_tau ~ dt(0,1,1)I(0,)
## [77]     beta1_sigma<-pow(1/beta1_tau,0.5)
## [78]
## [79]     #Overdispersion
## [80]     tauSigma ~ dunif(0,0.5)
## [81]     tauE <- pow(1/tauSigma,2)
## [82]
## [83]     #derived posterior check
## [84]     fit<-sum(E[]) #Discrepancy for the observed data
## [85]     fitnew<-sum(E.new[])
## [86]
## [87]     }
## [88]     ",fill=TRUE)
## [89]
## [90] sink()
```

```
  #for parallel run
  Yobs=indat$Yobs
  Bird=indat$jBird
  Plant=indat$jPlant
  Time=indat$jID
  Times=max(indat$jID)
  Traitmatch=jTraitmatch
  Birds=max(indat$jBird)
  Plants=max(indat$jPlant)
  Nobs=length(indat$Yobs)
  resources=resourcemat

  #A blank Y matrix - all present
  Ninit<-array(dim=c(Birds,Plants,Times),data=max(indat$Yobs)+1)

  #Inits
  InitStage <- function() {list(beta1=rep(0,Birds),alpha=rep(0,Birds),alpha_mu=0,N=Ninit,beta1_m
u=0,exp_lambda=Ninit)}

  #Parameters to track
  ParsStage <-
c("detect","alpha","beta1","alpha_mu","beta1_mu","fit","fitnew","ynew","E","tauE")

  #MCMC options
  ni <- runs  # number of draws from the posterior
  nt <- 10   #thinning rate
  nb <- max(0,runs-3000) # number to discard for burn-in
  nc <- 2  # number of chains

  Dat<-list("Yobs","Bird","Plant","Plants","Traitmatch","Birds","Nobs","Ninit","Time","Times","r
esources","nc","nb","ni","nt")

    system.time(abundance<-
jags.parallel(Dat,InitStage,parameters.to.save=ParsStage,model.file="Bayesian/NmixturePoissonRag
ged_Abundance.jags",n.thin=nt, n.iter=ni,n.burnin=nb,n.chains=nc))
```

```
##     user   system  elapsed
##    7.899    0.179 4029.018
```

```
#recompile if needed
load.module("dic")
runs<-100000
recompile(abundance)
abundance<-update(abundance,n.iter=runs,n.burnin=runs*.8,n.thin=5,parameters.to.save=ParsStage)
```

```
#extract par to data.frame
pars_abundance<-extract_par(abundance,data=indat,Bird="jBird",Plant="jPlant")
rm(abundance)
gc()
```

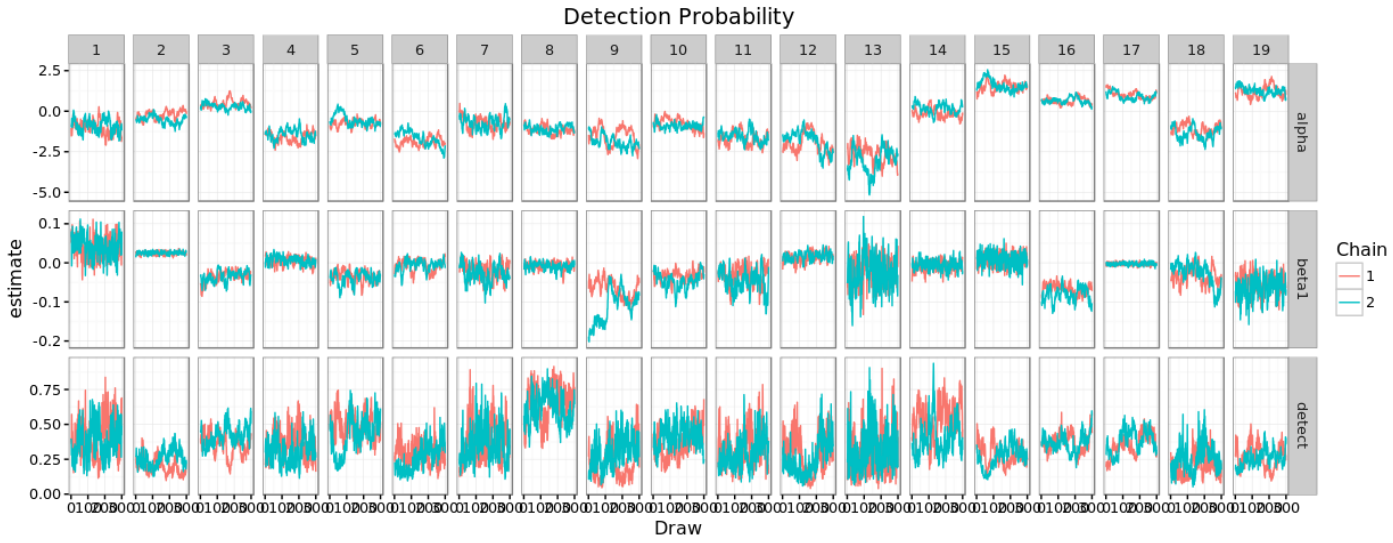
```
##             used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells   1688713  90.2   11615843  620.4  13178414  703.9
## Vcells 128780008 982.6  347018806 2647.6 344863298 2631.1
```

```
#name
pars_abundance$Model<-"N-mixture: Plant Abundance"
#write.csv(pars_abundance,"OutData/Abundance.csv")
```

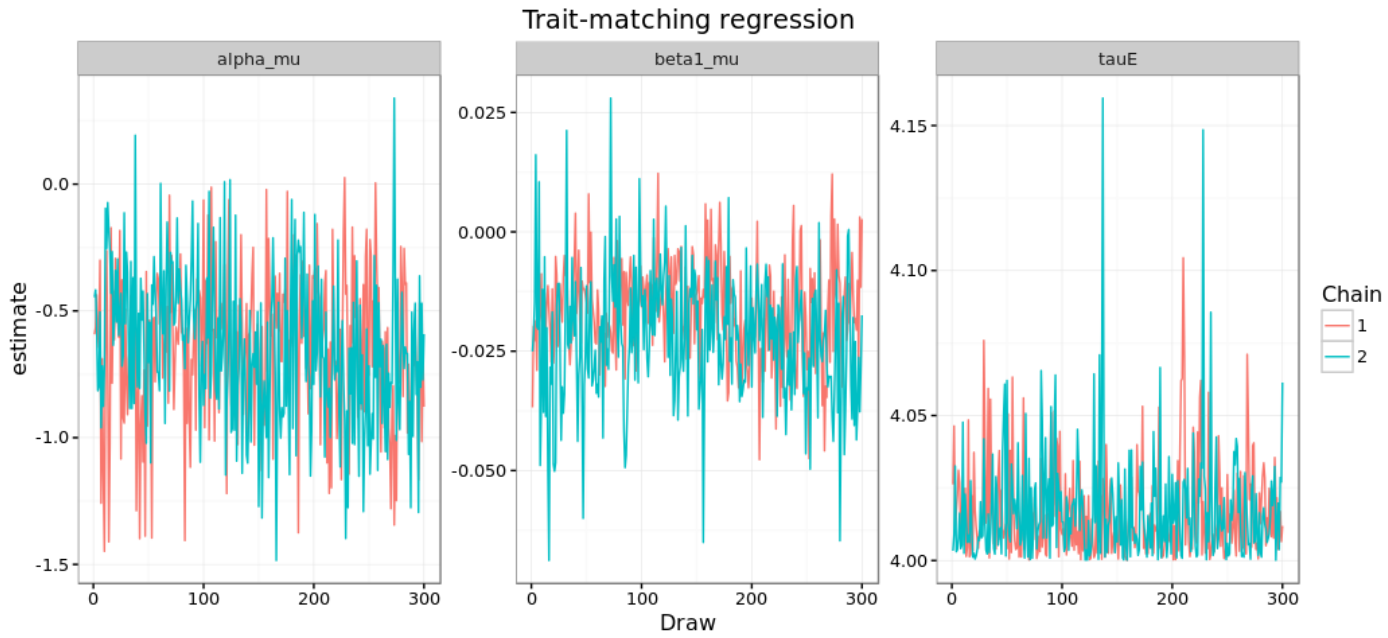
```
pars_abundance<-read.csv("OutData/Abundance.csv",row.names=1)
```

5.2.1 Assess Convergence

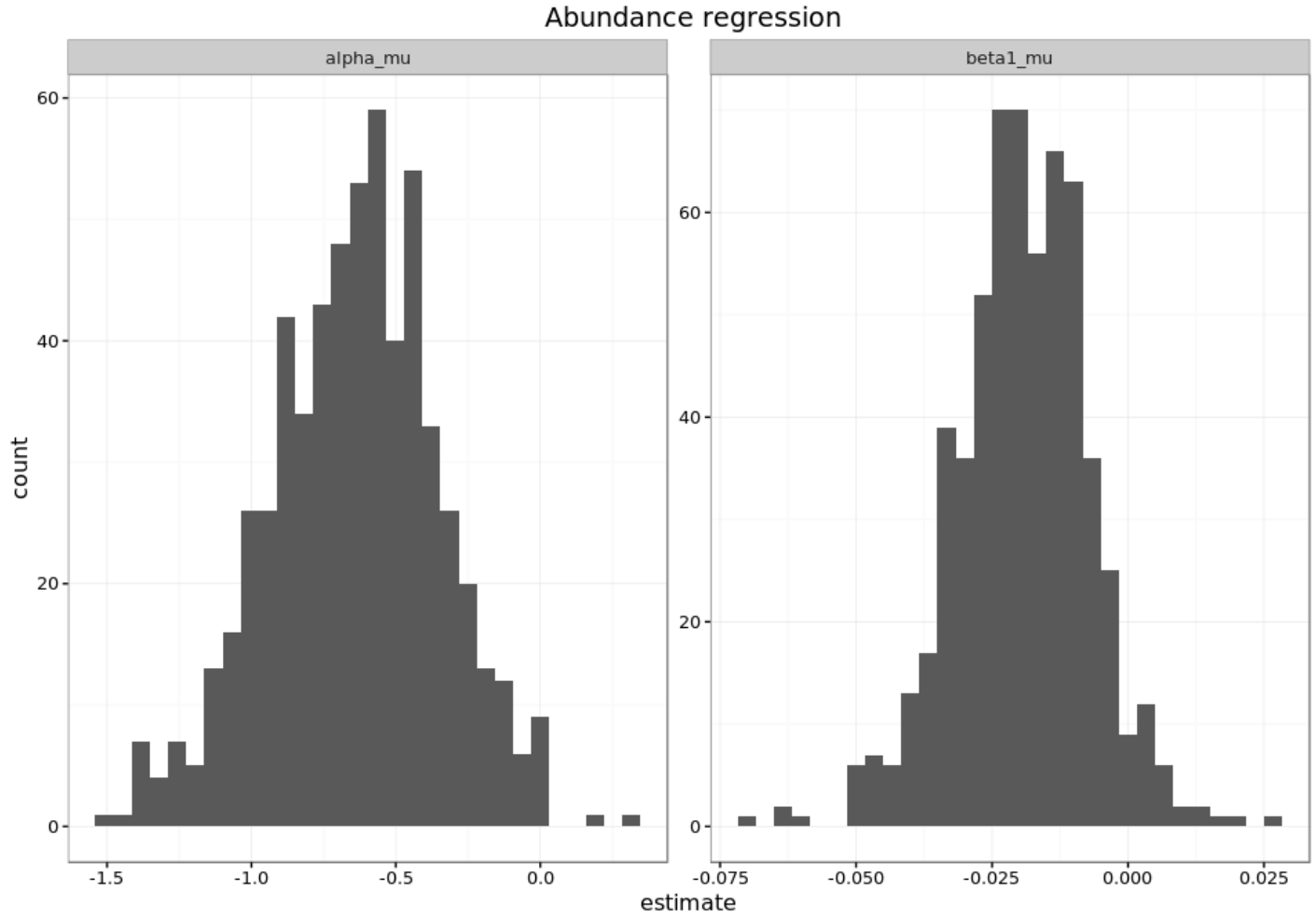
```
###Chains
ggplot(pars_abundance[pars_abundance$par %in% c("detect","alpha","beta1"),],aes(x=Draw,y=estimat
e,col=as.factor(Chain))) + geom_line() + facet_grid(par~species,scale="free") + theme_bw() + lab
s(col="Chain") + ggtitle("Detection Probability")
```




```
ggplot(pars_abundance[pars_abundance$par %in% c("beta1_mu","alpha_mu","sigma_alpha","beta1_sigm
a","sigma_detect","tauE"),],aes(x=Draw,y=estimate,col=as.factor(Chain))) + geom_line() + theme_b
w() + labs(col="Chain") + ggtitle("Trait-matching regression") + facet_wrap(~par,scales="free")
```



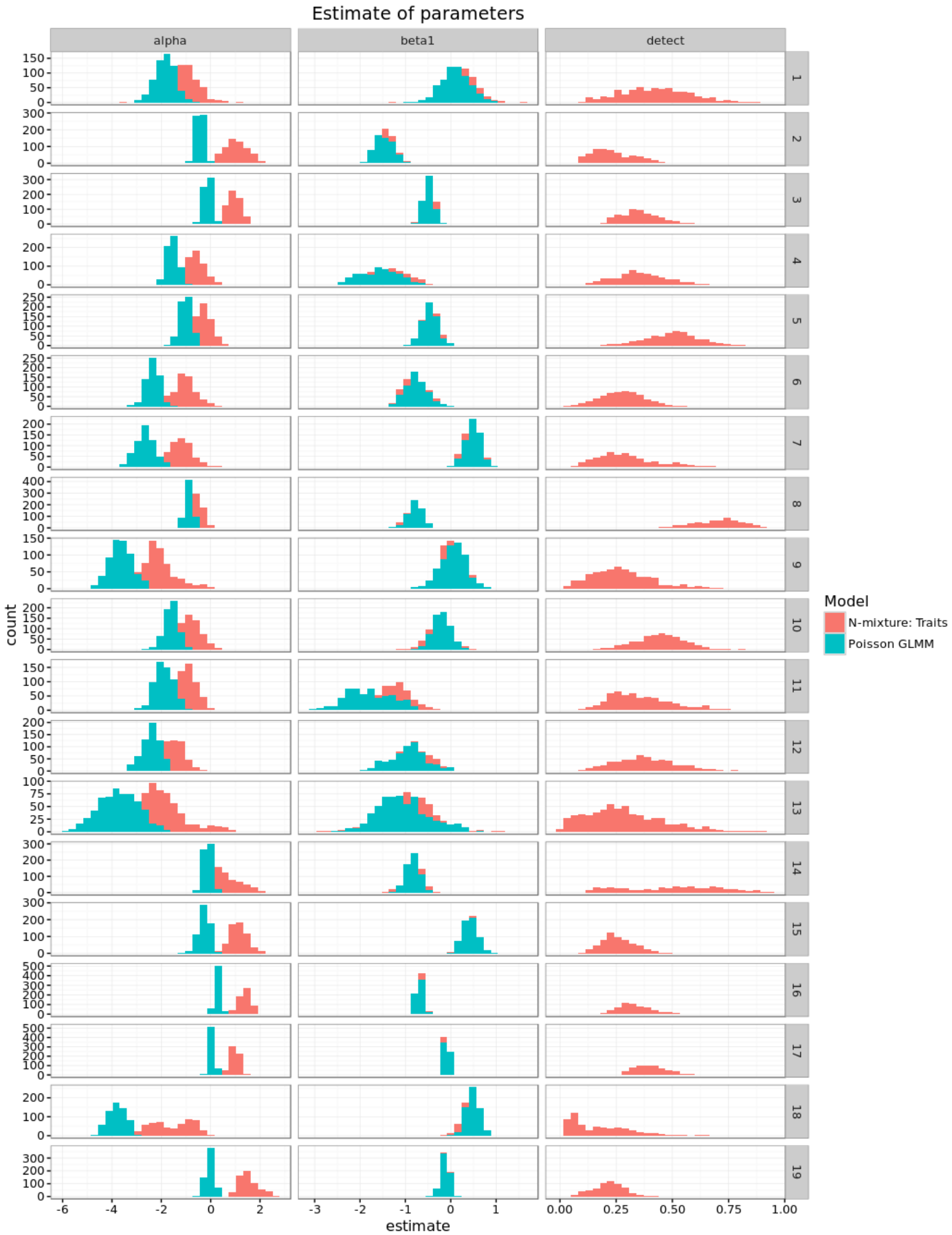

```
ggplot(pars_abundance[pars_abundance$par %in% c("beta1_mu","alpha_mu","sigma_alpha","beta1_sigm
a","sigma_detect"),],aes(x=estimate)) + geom_histogram() + theme_bw() + labs(col="Chain") + ggti
tle(" Abundance regression") + facet_wrap(~par,scales="free")
```



```
#Bind together the two models
parsk<-c("detect","alpha","beta1","tauE","beta1_mu","alpha_mu","sigma_alpha","beta1_sigma","fit","
itnew")

parsObs<-rbind(pars_detect_traits[pars_detect_traits$par %in% parsk,],pars_dniave[pars_dniave$pa
r %in% parsk,])
```

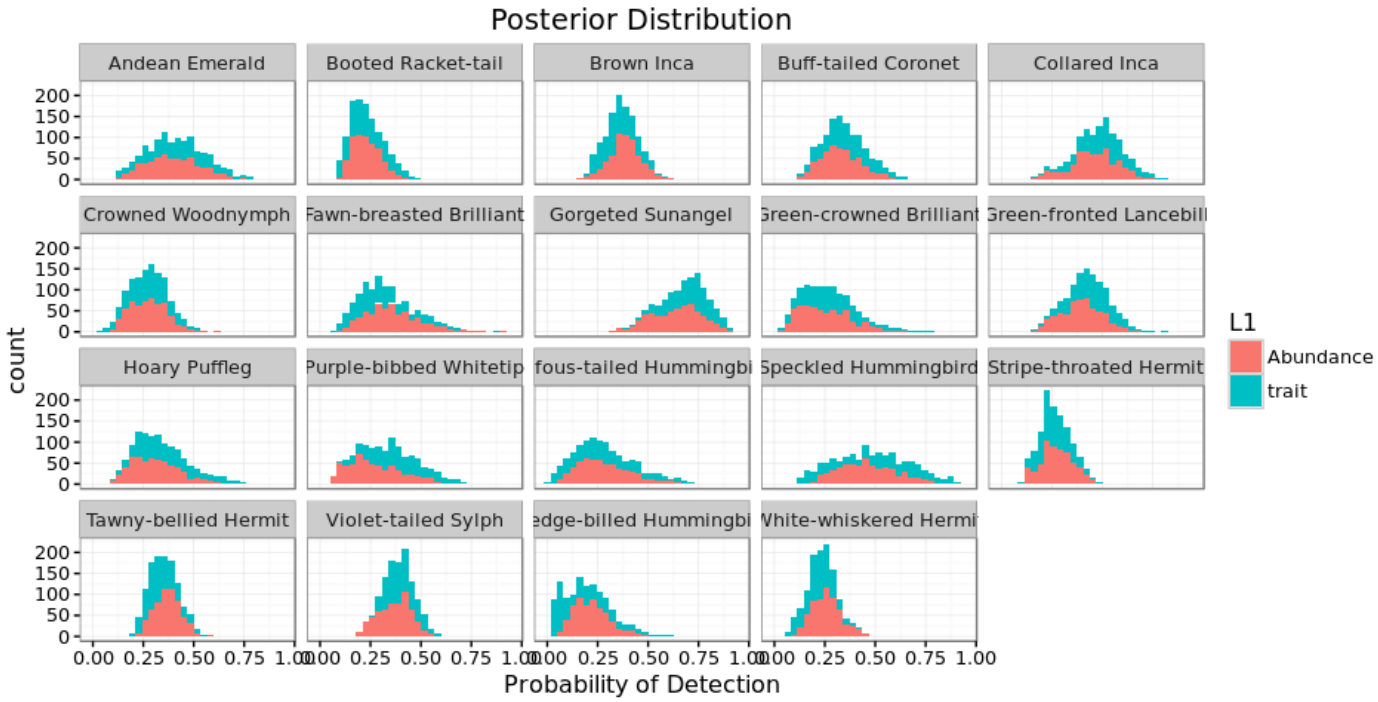
5.3 Posteriors

```
###Posterior Distributions
ggplot(parsObs[parsObs$par %in% c("detect","alpha","beta1"),],aes(x=estimate,fill=Model)) + geom
_histogram(position='identity') + ggtitle("Estimate of parameters") + facet_grid(species~par,sca
les="free") + theme_bw()
```



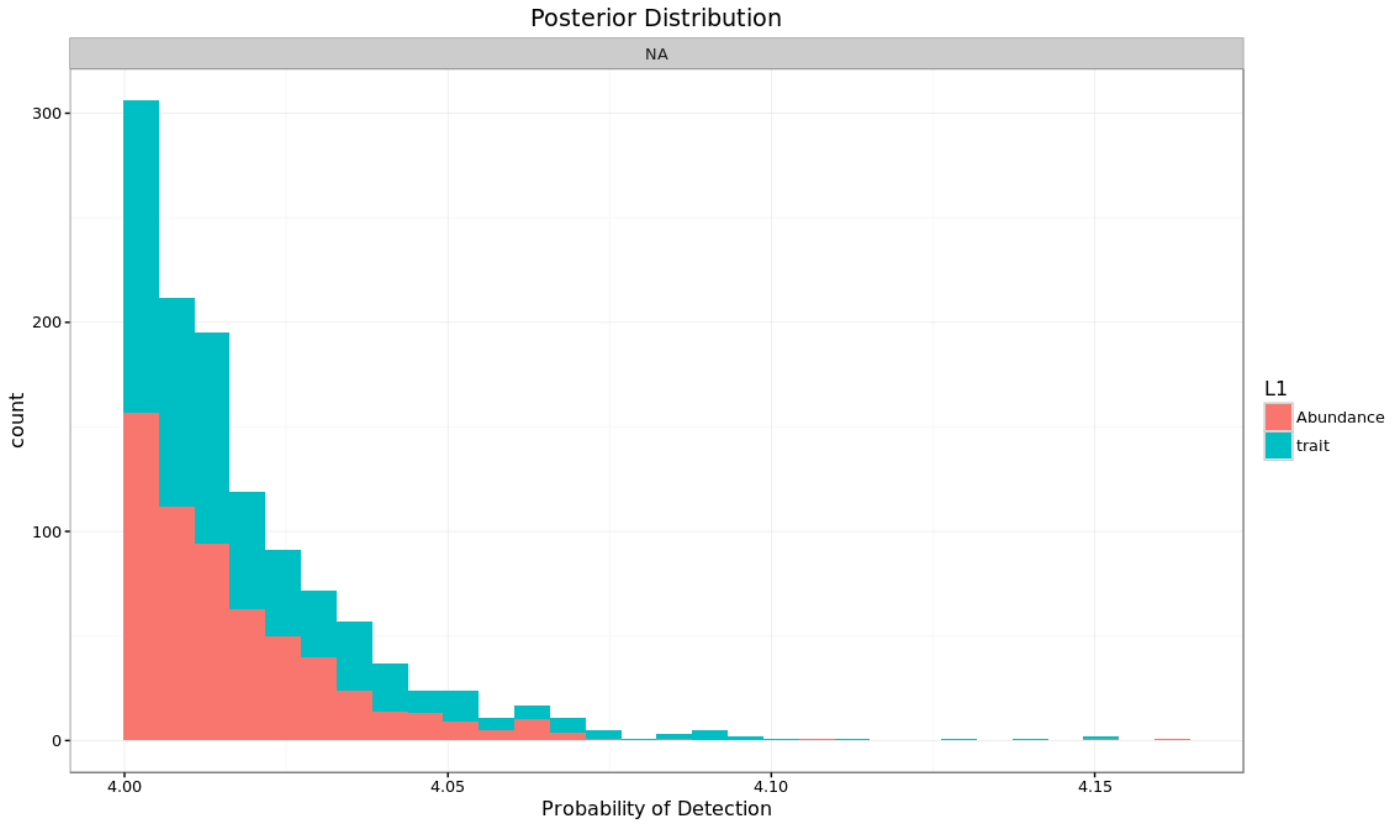
5.3.1 Parameter estimates in the trait and abundance N-mixture.

```
#Detection figure

#N-mixture models
par_at<-list(Abundance=pars_abundance %>% filter(par %in% c("detect")),trait=pars_detect_traits
%>% filter(par %in% c("detect")))
par_at<-melt(par_at,id.vars=colnames(pars_detect_traits))
par_at<-merge(par_at,jagsIndexBird,by.x="species",by.y="jBird",all.x=T)

ggplot(par_at,aes(x=estimate,fill=L1)) + geom_histogram() + ggtitle("Posterior Distribution") +
theme_bw() + facet_wrap(~Hummingbird,ncol=5) + xlab("Probability of Detection")
```



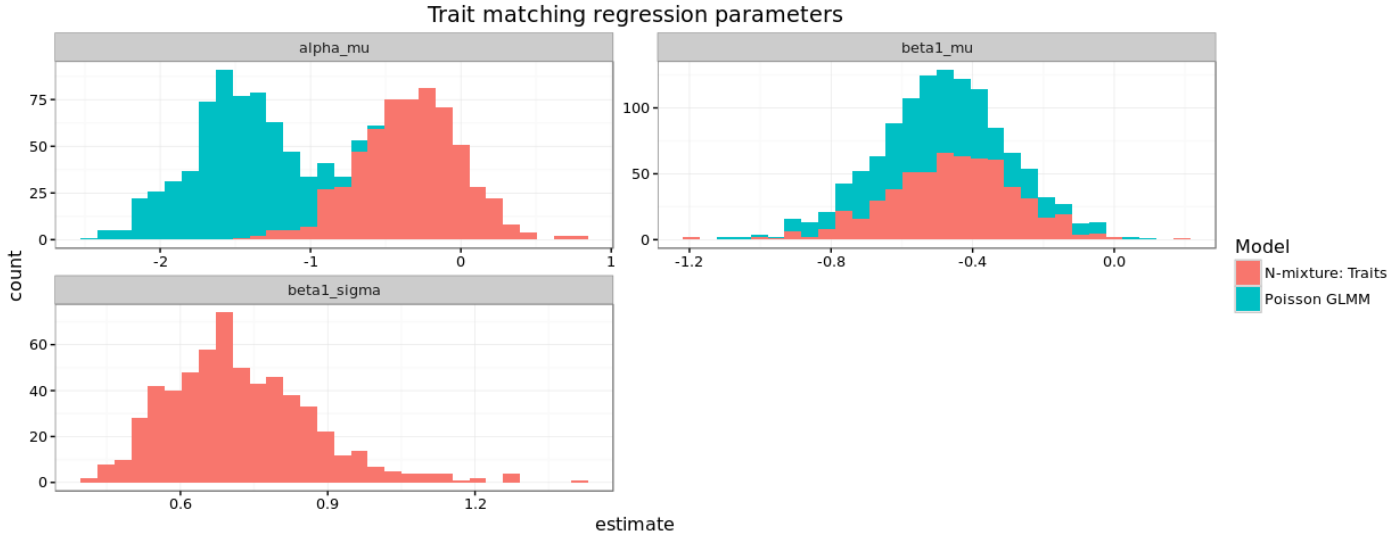

```
ggsave("Figures/DetectionProb.jpg",dpi=300,height=7,width=11)
```

```
#Detection figure

#N-mixture models
par_at<-list(Abundance=pars_abundance %>% filter(par %in% c("tauE")),trait=pars_detect_traits %
>% filter(par %in% c("tauE")))
par_at<-melt(par_at,id.vars=colnames(pars_detect_traits))
par_at<-merge(par_at,jagsIndexBird,by.x="species",by.y="jBird",all.x=T)

ggplot(par_at,aes(x=estimate,fill=L1)) + geom_histogram() + ggtitle("Posterior Distribution") +
theme_bw() + facet_wrap(~Hummingbird,ncol=5) + xlab("Probability of Detection")
```




```
ggsave("Figures/DetectionProb.jpg",dpi=300,height=7,width=11)
```

```
ggplot(parsObs[parsObs$par %in% c("beta1_mu","alpha_mu","sigma_alpha","beta1_sigma"),],aes(x=est
imate,fill=Model)) + geom_histogram() + ggtitle("Trait matching regression parameters") + facet_
wrap(~par,scale="free",nrow=2) + theme_bw()
```

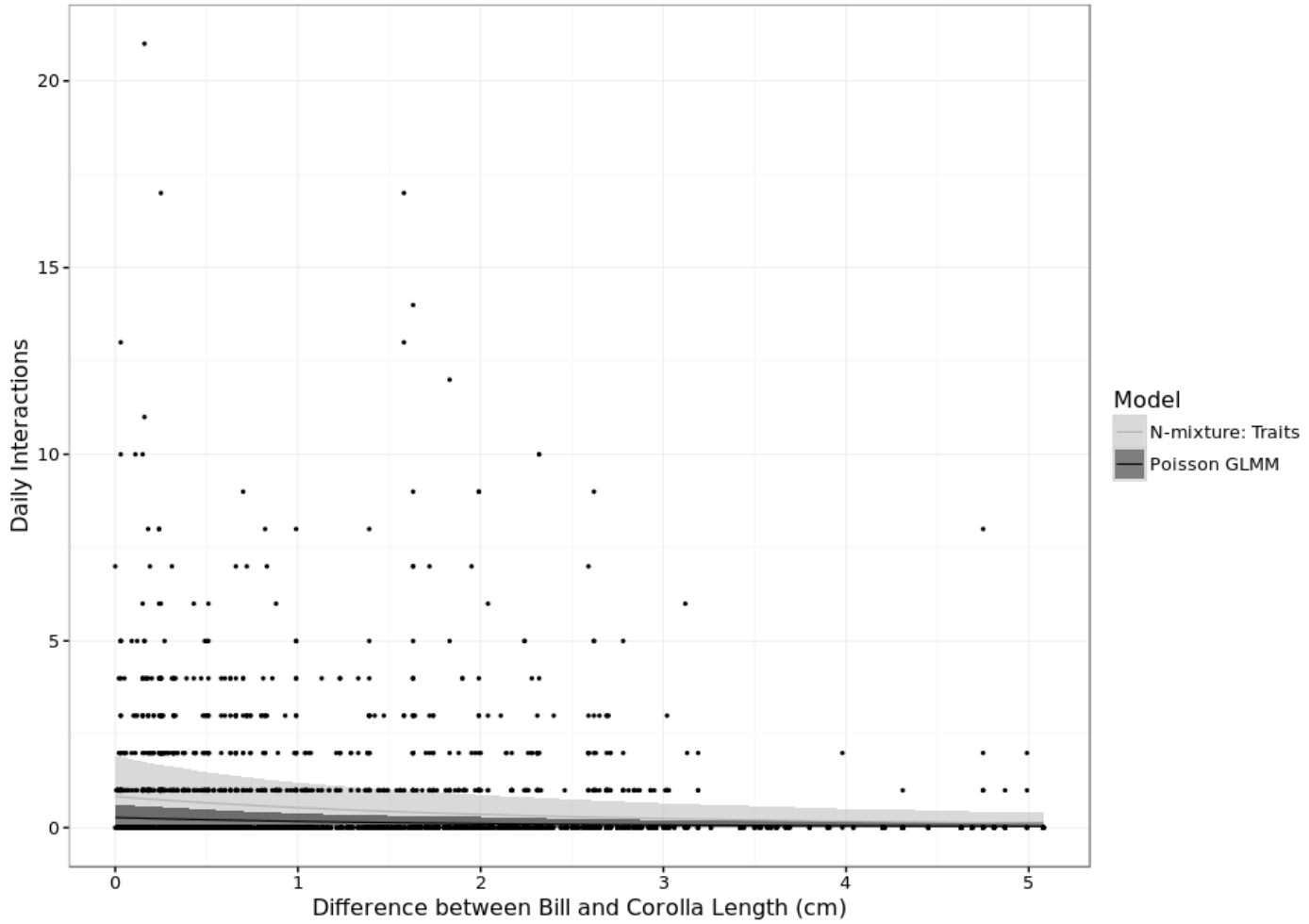


5.3.2 Predicted relationship

5.4 Poisson GLMM versus N-mixture model

```
castdf<-dcast(parsObs[parsObs$par %in% c("beta1_mu","alpha_mu","tauE"),], Model+Chain + Draw~par,v
lue.var="estimate")

castdf<-split(castdf,castdf$Model)
```

```
predy<-rbind_all(lapply(castdf,function(i){
  #calculate trajectory and append model
  pr<-trajF(alpha=i$alpha_mu,beta1=i$beta1_mu,tauSigma=1/sqrt(i$tauE),trait=indat$Traitmatch,res
ources=indat$scaledR)
  pr$Model<-unique(i$Model)
  return(pr)
  }))

fplot<-ggplot(data=predy[,],aes(x=trait)) + geom_ribbon(aes(ymin=lower,ymax=upper,fill=Model),al
pha=0.5)  + geom_line(aes(y=mean,col=Model),size=.4) + theme_bw() + ylab("Daily Interactions") +
 xlab("Difference between Bill and Corolla Length (cm)") + geom_point(data=indat,aes(x=Traitmatc
h,y=Yobs),size=.5,alpha=1) + labs(fill="Model",col="Model")  + scale_fill_manual(values=c("grey7
0","black"))+ scale_color_manual(values=c("grey70","black"))
fplot
```



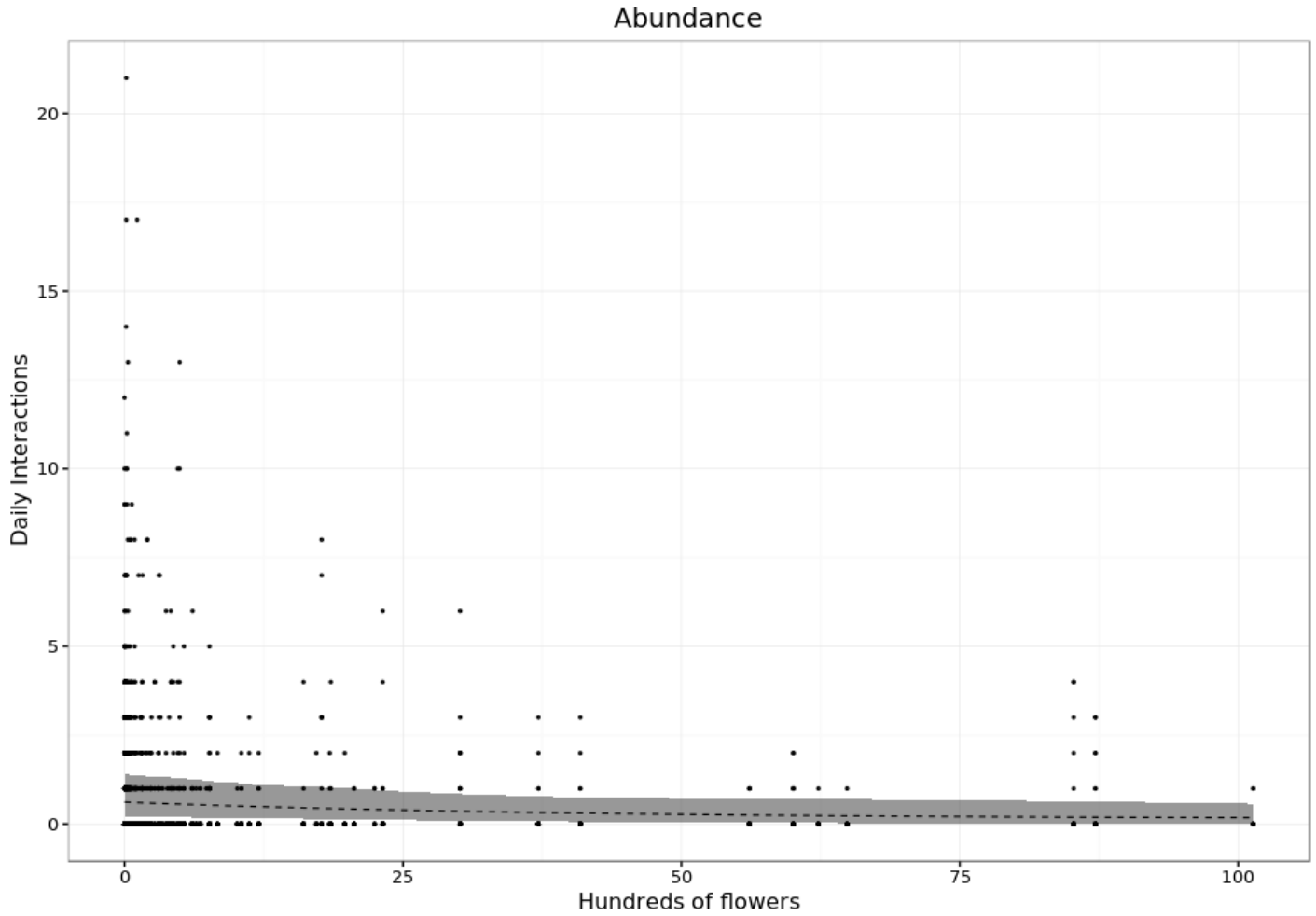

```
ggsave("Figures/BothObs.jpg",heigh=5,width=7,dpi=300)
```

5.5 Traits

```
castdf<-dcast(pars_detect_traits[pars_detect_traits$par %in% c("beta1_mu","alpha_mu","tauE"),],
Chain + Draw~par,value.var="estimate")

predy_traits<-trajF(alpha=castdf$alpha_mu,beta1=castdf$beta1_mu,tauSigma=1/sqrt(castdf$tauE),tra
it=indat$Traitmatch,resources=indat$scaledR)

tplot<-ggplot(data=predy_traits,aes(x=trait)) +
geom_ribbon(aes(ymin=lower,ymax=upper),alpha=0.5)  + geom_line(aes(y=mean),size=.4,linetype="das
hed") + theme_bw() + ylab("Daily Interactions") + xlab("Difference between Bill and Corolla Leng
th (cm)") + geom_point(data=indat,aes(x=Traitmatch,y=Yobs),size=.5,alpha=1) + ggtitle("Traits")
tplot
```

5.6 Abundance

```
castdf<-dcast(pars_abundance[pars_abundance$par %in% c("beta1_mu","alpha_mu","tauE"),], Chain +
Draw~par,value.var="estimate")

predy_abundance<-
trajF(alpha=castdf$alpha_mu,beta1=castdf$beta1_mu,tauSigma=1/sqrt(castdf$tauE),trait=indat$scale
dR)

aplot<-ggplot(data=predy_abundance,aes(x=trait)) + geom_ribbon(aes(ymin=lower,ymax=upper),alpha=0.
  + geom_line(aes(y=mean),size=.4,linetype="dashed") + theme_bw() + ylab("Daily Interactions") +
 xlab("Hundreds of flowers") + geom_point(data=indat,aes(x=scaledR,y=Yobs),size=.5,alpha=.9) + g
gtitle("Abundance")
aplot
```



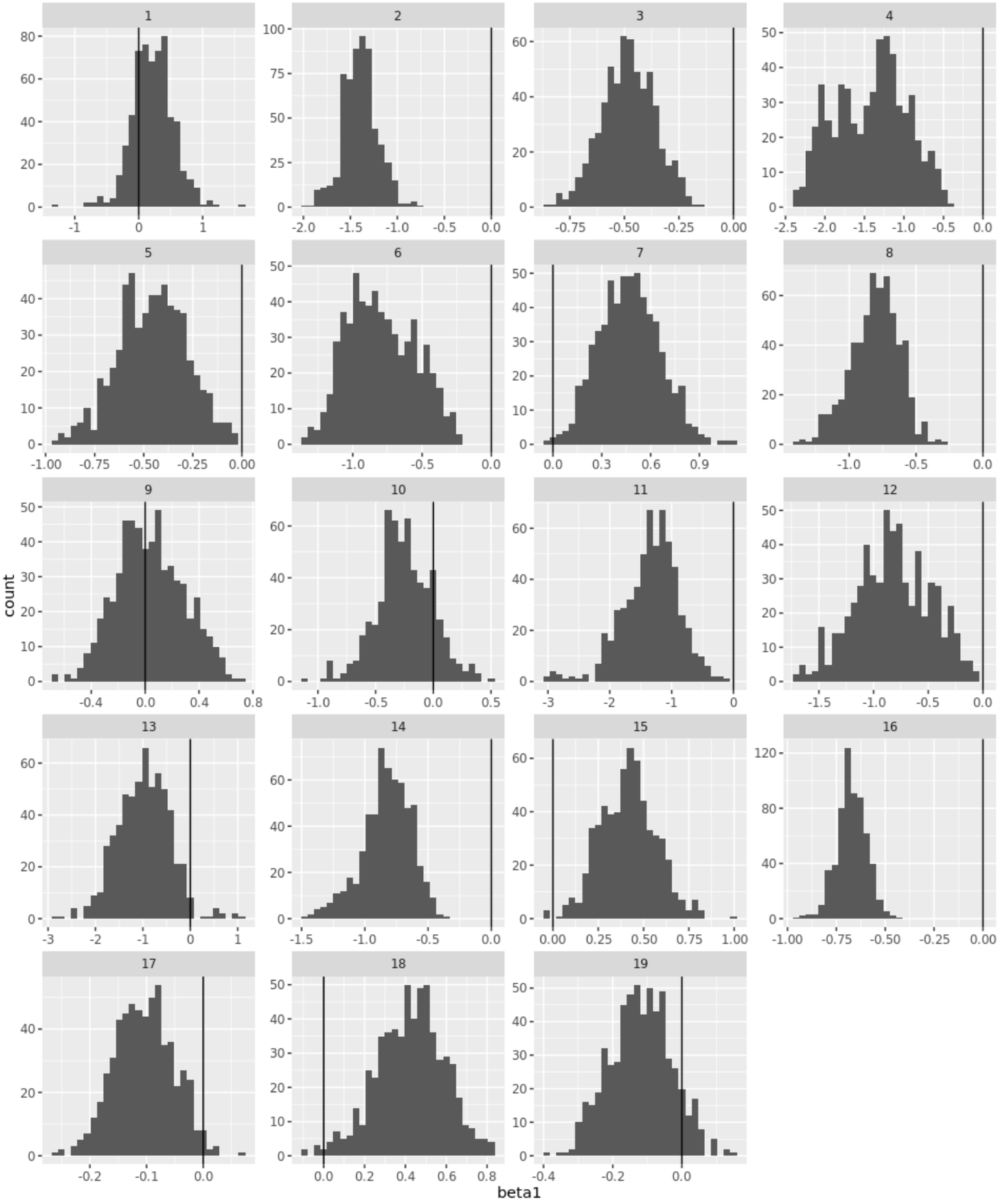

```
ggsave("Figures/AbundanceBothPlot.jpeg",height=4,width=7,dpi=300)
```

6 Species Predictions

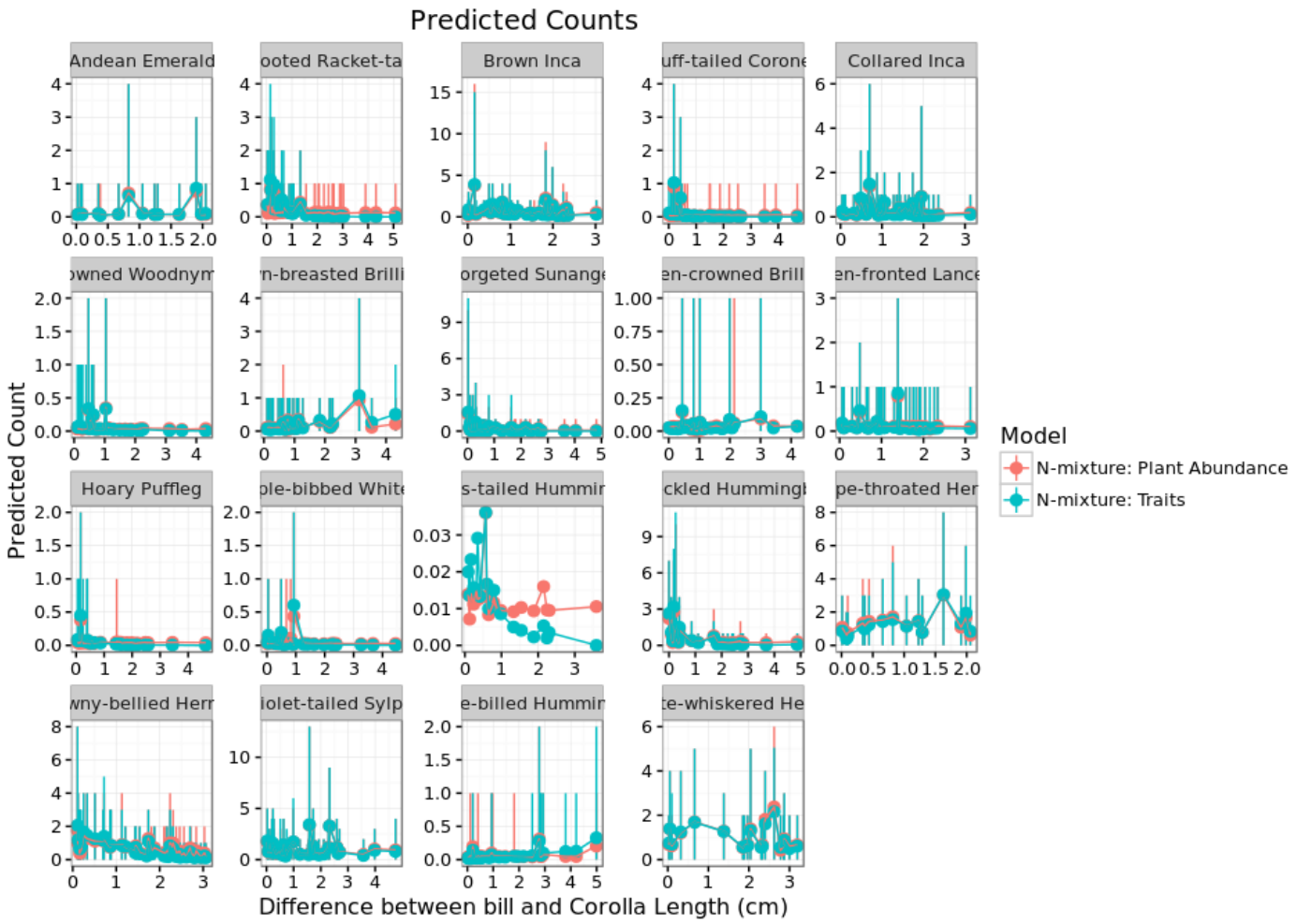
6.1 Credible intervals

6.1.1 Traits

```
castdf<-dcast(pars_detect_traits[pars_detect_traits$par %in% c("beta1","alpha"),], species +Chai
n +Model+ Draw~par,value.var="estimate")

ggplot(castdf,aes(x=beta1)) + geom_histogram() + facet_wrap(~species,scales="free",ncol=4) + geo
m_vline(xintercept=0)
```

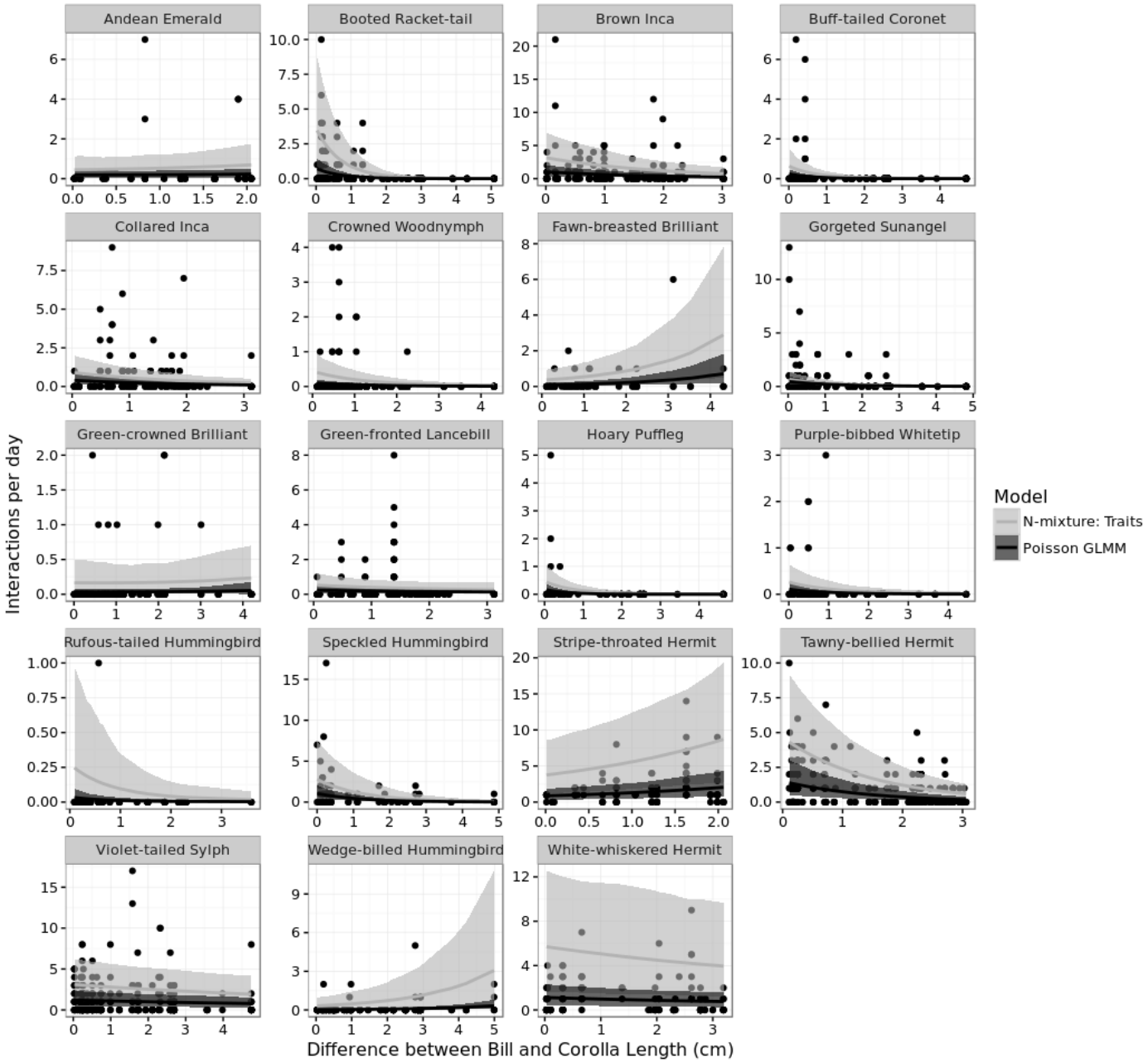

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
count
beta1

```
par_at<-list(Abundance=pars_abundance %>% filter(par %in% c("ynew")),trait=pars_detect_traits %
>% filter(par %in% c("ynew")))
par_at<-melt(par_at,id.vars=colnames(pars_detect_traits))
ynewstat<-dcast(par_at,Model+Index+Draw+Chain~par,value.var="estimate")

#By species
ynewstat$Hummingbird<-indat[ynewstat$Index,"Hummingbird"]
ynewstat$Traitmatch<-indat[ynewstat$Index,"Traitmatch"]

mean_ynewstat<-ynewstat %>% group_by(Hummingbird,Model,Traitmatch) %>% summarize(mean=mean(ynew),l
wer=quantile(ynew,0.05),upper=quantile(ynew,0.95))

ggplot(mean_ynewstat,aes(x=Traitmatch,col=Model)) + geom_pointrange(aes(y=mean,ymin=lower,ymax=u
pper)) + labs(x="Difference between bill and Corolla Length (cm)",y="Predicted Count") + ggtitle("
redicted Counts") + theme_bw() + facet_wrap(~Hummingbird,scales="free") + geom_line(aes(y=mean))
```



6.2 Prediction intervals

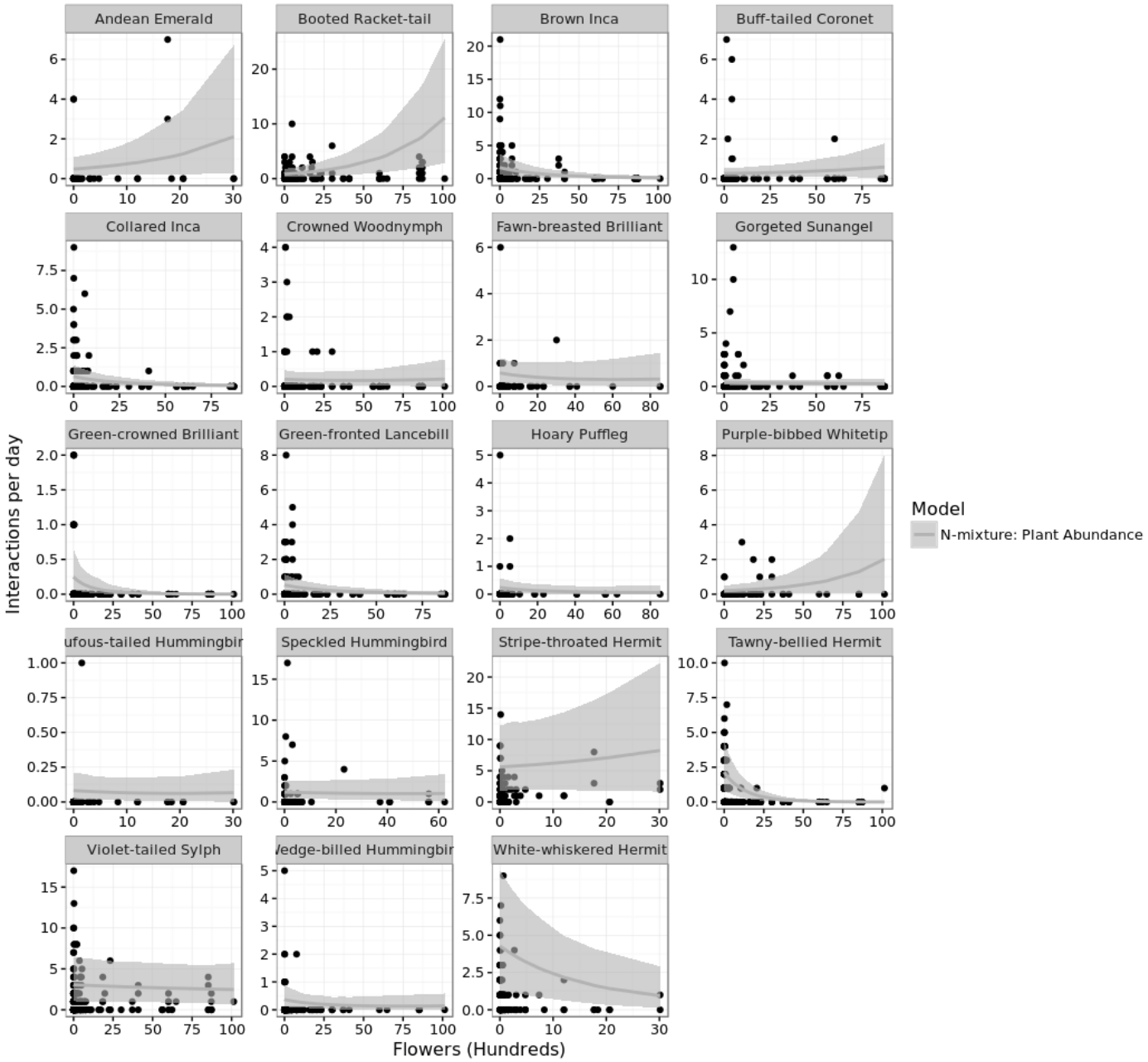
```
castdf<-dcast(parsObs[parsObs$par %in% c("beta1","alpha"),], species +Chain +Model+ Draw~par,val
ue.var="estimate")

tcastdf<-dcast(parsObs[parsObs$par %in% c("tauE"),],  Chain +Model+ Draw~par,value.var="estimat
e")

castdf<-merge(castdf,tcastdf,by=c("Chain","Model","Draw"))

#Turn to species level
castdf$species<-factor(castdf$species,levels=1:max(as.numeric(castdf$species)))

species.split<-split(castdf,list(castdf$species,castdf$Model),drop=T)

species.traj<-list()

for(d in 1:length(species.split)){
  x<-species.split[[d]]
  #species name
  index<-jagsIndexBird[unique(x$species),"Hummingbird"]

  #range of trait distances
  tsp<-indat %>% filter(Hummingbird==index) %>% .$Traitmatch

  #Range of abundances
  fsp<-indat %>% filter(Hummingbird==index) %>% .$scaledR

  species.traj[[d]]<-trajF(alpha=x$alpha,beta1=x$beta1,tauSigma=1/sqrt(x$tauE),trait=tsp,resourc
es=fsp)
}

names(species.traj)<-names(species.split)

species.traj<-melt(species.traj,id.var=colnames(species.traj[[1]]))

#split out names and model
species.traj[,c("Index","Model")]<-colsplit(species.traj$L1,"\\.",c("Index","Model"))

spe<-merge(species.traj,jagsIndexBird,by.x="Index",by.y="jBird")

#match colnames

#plot and compare to original data
ggplot(data=spe[,],aes(x=trait)) + geom_point(data=indat,aes(x=Traitmatch,y=Yobs)) + geom_ribbon(a
s(ymin=lower,ymax=upper,fill=Model),alpha=0.6)  + geom_line(aes(y=mean,col=Model),size=1) + them
e_bw() + ylab("Interactions") + xlab("Difference between Bill and Corolla Length (cm)") + facet_
wrap(~Hummingbird,scales="free",ncol=4)+ labs(fill="Model")  + ylab("Interactions per day") + sc
ale_color_manual(values=c("grey70","black")) + scale_fill_manual(values=c("grey70","black"))
```




```
ggsave("Figures/SpeciesPredictionsBoth.jpg",dpi=300,height=9,width=11)
```

6.2.1 Abundance Species predictions

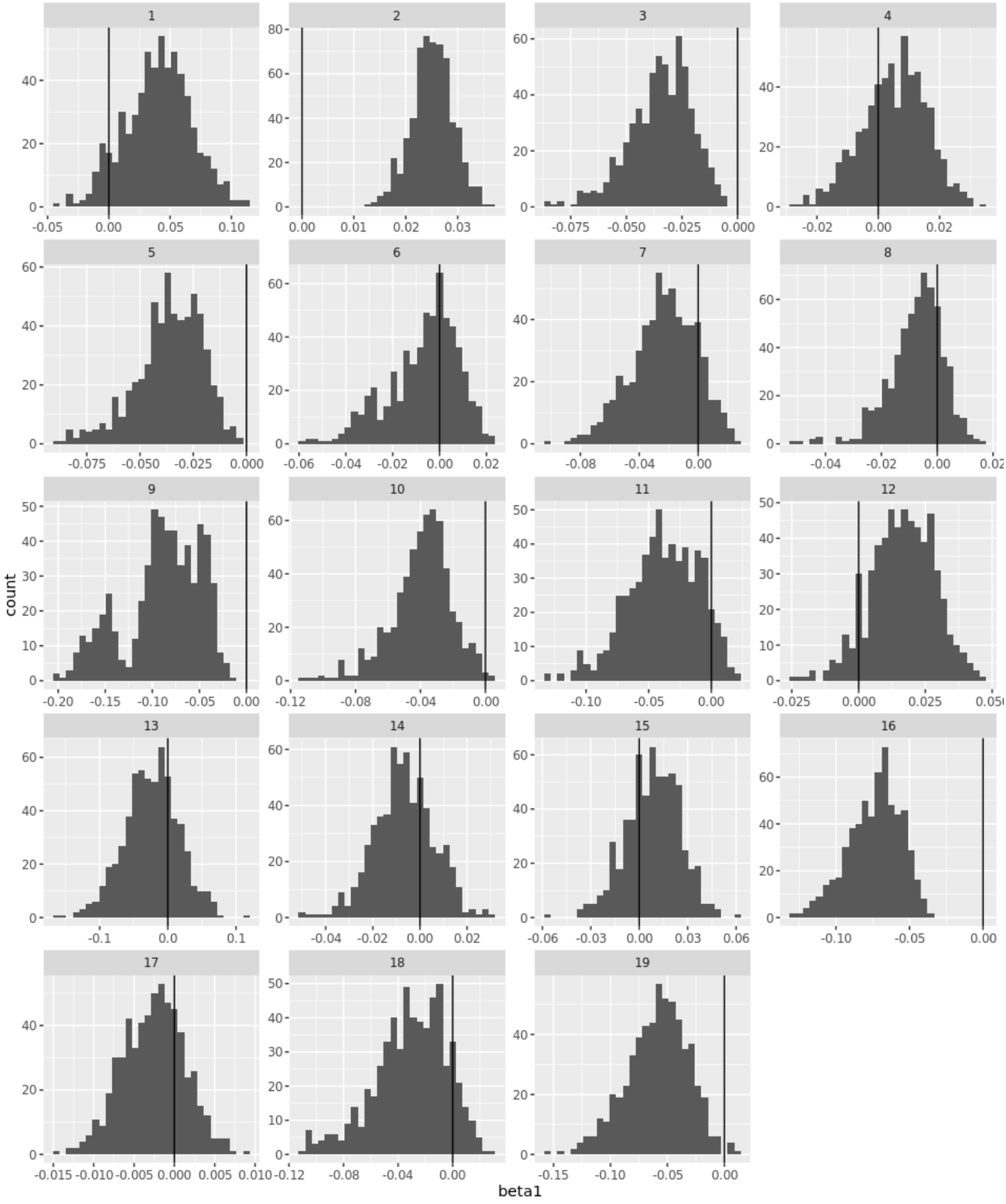
```
castdf<-dcast(pars_abundance[pars_abundance$par %in% c("beta1","alpha"),], species +Chain
+Model+ Draw~par,value.var="estimate")

tcastdf<-dcast(pars_abundance[pars_abundance$par %in% c("tauE"),],  Chain +Model+ Draw~par,valu
e.var="estimate")

castdf<-merge(castdf,tcastdf,by=c("Chain","Model","Draw"))

#Turn to species level
castdf$species<-factor(castdf$species,levels=1:max(as.numeric(castdf$species)))

species.split<-split(castdf,list(castdf$species,castdf$Model))

species.traj<-list()

for(d in 1:length(species.split)){
  x<-species.split[[d]]
  #species name
  index<-jagsIndexBird[unique(x$species),"Hummingbird"]

  #range of trait distances
  tsp<-indat %>% filter(Hummingbird==index) %>% .$scaledR

  #Range of abundances
    fsp<-indat %>% filter(Hummingbird==index) %>% .$scaledR

  species.traj[[d]]<-trajF(alpha=x$alpha,beta1=x$beta1,tauSigma=1/sqrt(x$tauE),trait=tsp,resourc
es=fsp)
}

names(species.traj)<-names(species.split)

species.traj<-melt(species.traj,id.var=colnames(species.traj[[1]]))

#split out names and model
species.traj[,c("Index","Model")]<-colsplit(species.traj$L1,"\\.",c("Index","Model"))

spe<-merge(species.traj,jagsIndexBird,by.x="Index",by.y="jBird")

#match colnames

#plot and compare to original data
ggplot(data=spe[,],aes(x=trait)) + geom_point(data=indat,aes(x=scaledR,y=Yobs)) + geom_ribbon(ae
s(ymin=lower,ymax=upper,fill=Model),alpha=0.6)  + geom_line(aes(y=mean,col=Model),size=1) + them
e_bw() + ylab("Interactions") + xlab("Flowers (Hundreds)") + facet_wrap(~Hummingbird,scales="fre
e",ncol=4)+ labs(fill="Model")  + ylab("Interactions per day") + scale_color_manual(values=c("gr
ey70","black")) + scale_fill_manual(values=c("grey70","black"))
```




```
ggsave("Figures/SpeciesPredictions_Abundance.jpg",dpi=300,height=10,width=10)
```

6.3 Abundance posteriors

```
castdf<-dcast(pars_abundance[pars_abundance$par %in% c("beta1","alpha"),], species +Chain
+Model+ Draw~par,value.var="estimate")

ggplot(castdf,aes(x=beta1)) + geom_histogram() + facet_wrap(~species,scales="free",ncol=4) + geo
m_vline(xintercept=0)
```

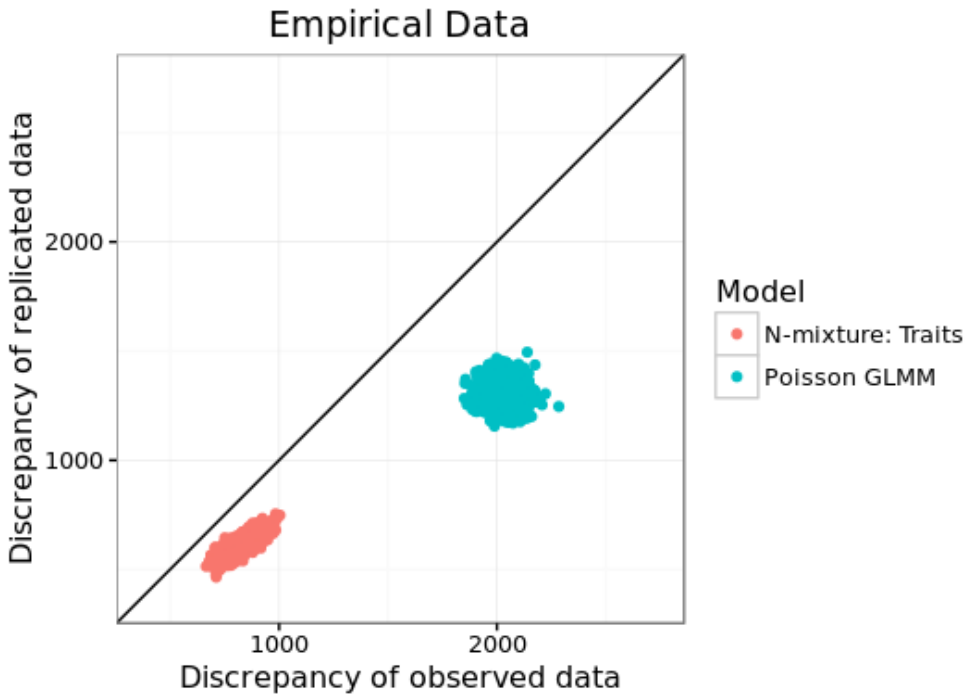


7 Discrepancy: N-mixture v Poisson GLM

The goodness of fit is a measured as chi-squared. The expected value for each day is the detection rate * the estimate intensity of interactions. The expected value is compared to the observed value of the actual data. In addition, a replicate dataset is generated from the posterior predicted intensity. Better fitting models will

have lower discrepancy values and be Better fitting models are smaller values and closer to the 1:1 line. A perfect model would be 0 discrepancy. This is unrealsitic given the stochasticity in the sampling processes. Rather, its better to focus on relative discrepancy. In addition, a model with 0 discrepancy would likely be seriously overfit and have little to no predictive power.

```
fitstat<-parsObs[parsObs$par %in% c("fit","fitnew"),]
fitstat<-dcast(fitstat,Model+Draw+Chain~par,value.var="estimate")

ymin<-min(c(fitstat$fit,fitstat$fitnew)) - min(c(fitstat$fit,fitstat$fitnew)) * .2
ymax<-max(c(fitstat$fit,fitstat$fitnew)) + max(c(fitstat$fit,fitstat$fitnew)) * .2
disc_obs<-ggplot(fitstat,aes(x=fit,y=fitnew)) + geom_point(aes(col=Model)) + theme_bw() + labs(x="
iscrepancy of observed data",y="Discrepancy of replicated data",col="Model")  + ggtitle("Empiric
al Data") + geom_abline() + coord_fixed() + ylim(ymin=ymin,ymax=ymax) +
xlim(xmin=ymin,xmax=ymax)
disc_obs
```



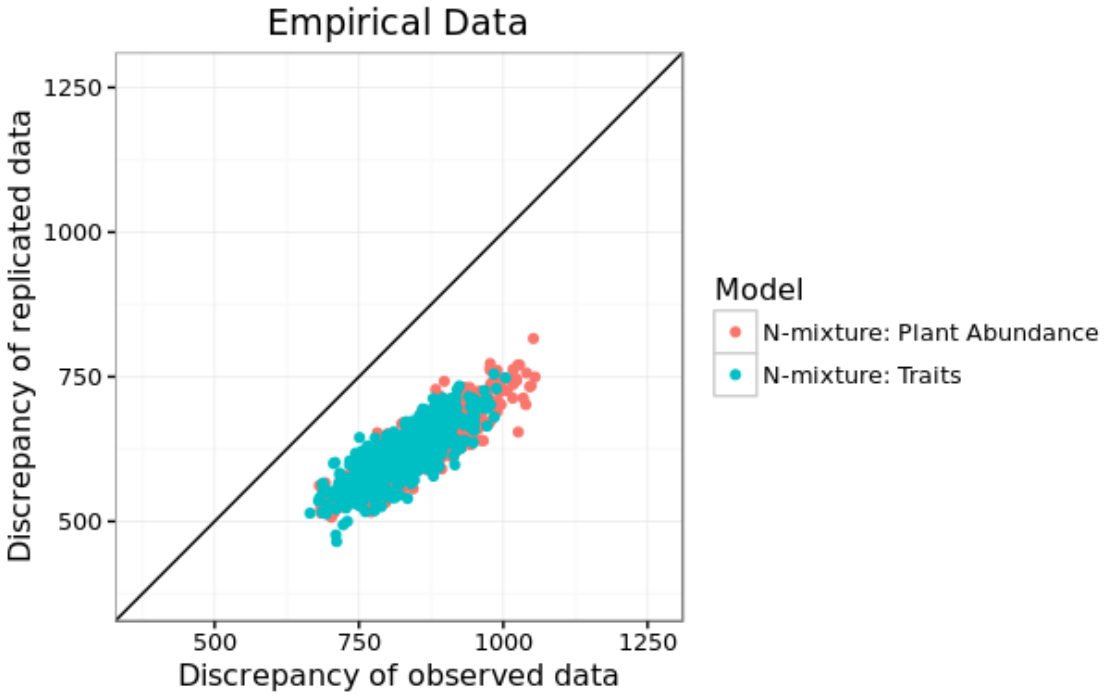

```
fitstat %>% group_by(Model) %>% summarize(mean(fit),sum(fit))
```

```
## Source: local data frame [2 x 3]
##
##               Model mean(fit)  sum(fit)
##               (chr)     (dbl)     (dbl)
## 1 N-mixture: Traits  816.5002  489900.1
## 2      Poisson GLMM 2022.9839 1213790.3
```

```
ggsave("Figures/ObservedDiscrepancy.jpeg",width = 5,height=10)
```

8 Discrepancy: Abundance v Traits

```
par_at<-list(Abundance=pars_abundance %>% filter(par %in% c("fit","fitnew")),trait=pars_detect_t
raits %>% filter(par %in% c("fit","fitnew")))
par_at<-melt(par_at,id.vars=colnames(pars_detect_traits))

fitstat<-dcast(par_at,Model+Draw+Chain~par,value.var="estimate")

ymin<-min(c(fitstat$fit,fitstat$fitnew)) - min(c(fitstat$fit,fitstat$fitnew)) * .2
ymax<-max(c(fitstat$fit,fitstat$fitnew)) + max(c(fitstat$fit,fitstat$fitnew)) * .2
disc_obs<-ggplot(fitstat,aes(x=fit,y=fitnew)) + geom_point(aes(col=Model)) + theme_bw() + labs(x="
iscrepancy of observed data",y="Discrepancy of replicated data",col="Model")  + ggtitle("Empiric
al Data") + geom_abline() + coord_fixed() + ylim(ymin=ymin,ymax=ymax) +
xlim(xmin=ymin,xmax=ymax)
disc_obs
```



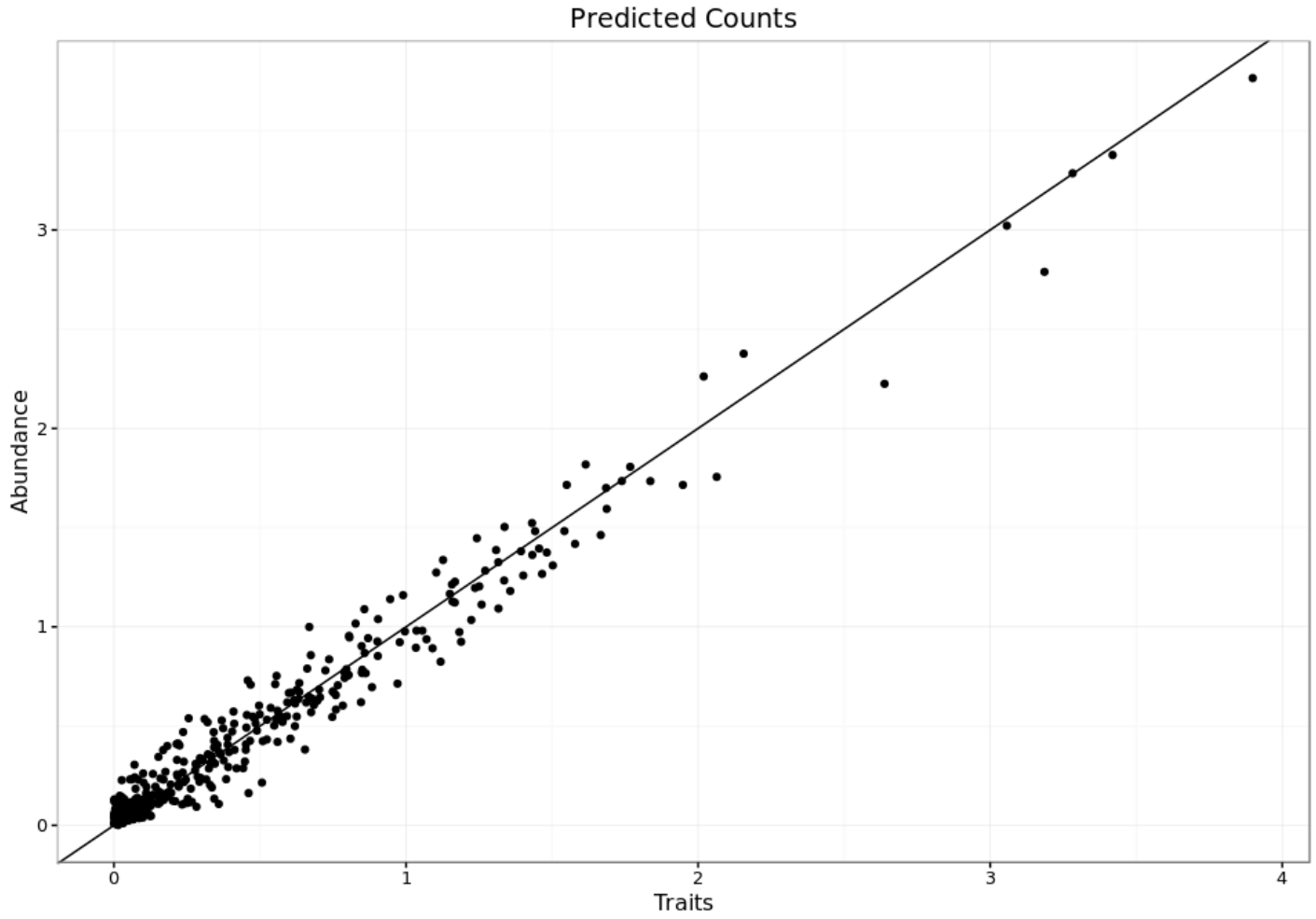

```
fitstat %>% group_by(Model) %>% summarize(mean(fit),sum(fit))
```

```
## Source: local data frame [2 x 3]
##
##                         Model mean(fit) sum(fit)
##                         (chr)     (dbl)    (dbl)
## 1 N-mixture: Plant Abundance  858.0223 514813.4
## 2          N-mixture: Traits  816.5002 489900.1
```

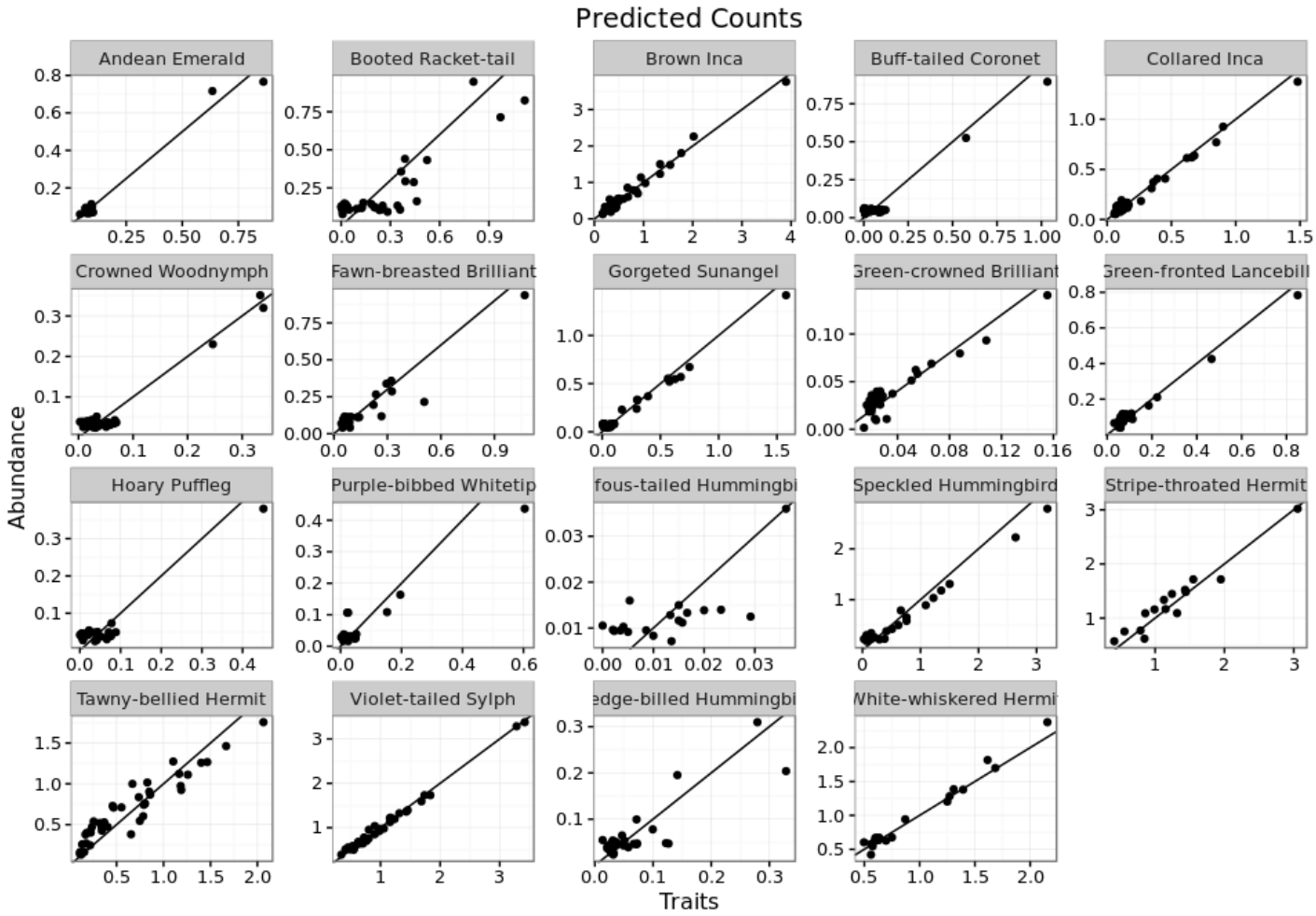
```
ggsave("Figures/ATDiscrepancy.jpeg",width = 5,height=10)
```

8.0.1 Correlation in prediction

```
par_at<-list(Abundance=pars_abundance %>% filter(par %in% c("ynew")) %>% group_by(species,plant)
 %>% summarize(estimate=mean(estimate)) ,trait=pars_detect_traits %>% filter(par %in% c("ynew"))
 %>% group_by(species,plant) %>% summarize(estimate=mean(estimate)))
par_at<-melt(par_at,id.vars=c("species","plant","estimate"))
ynewstat<-dcast(par_at,...~L1,value.var="estimate")

#plot
ggplot(ynewstat,aes(x=trait,y=Abundance)) + geom_point() + geom_abline() + labs(x="Traits",y="Ab
undance") + ggtitle("Predicted Counts") + theme_bw()
```



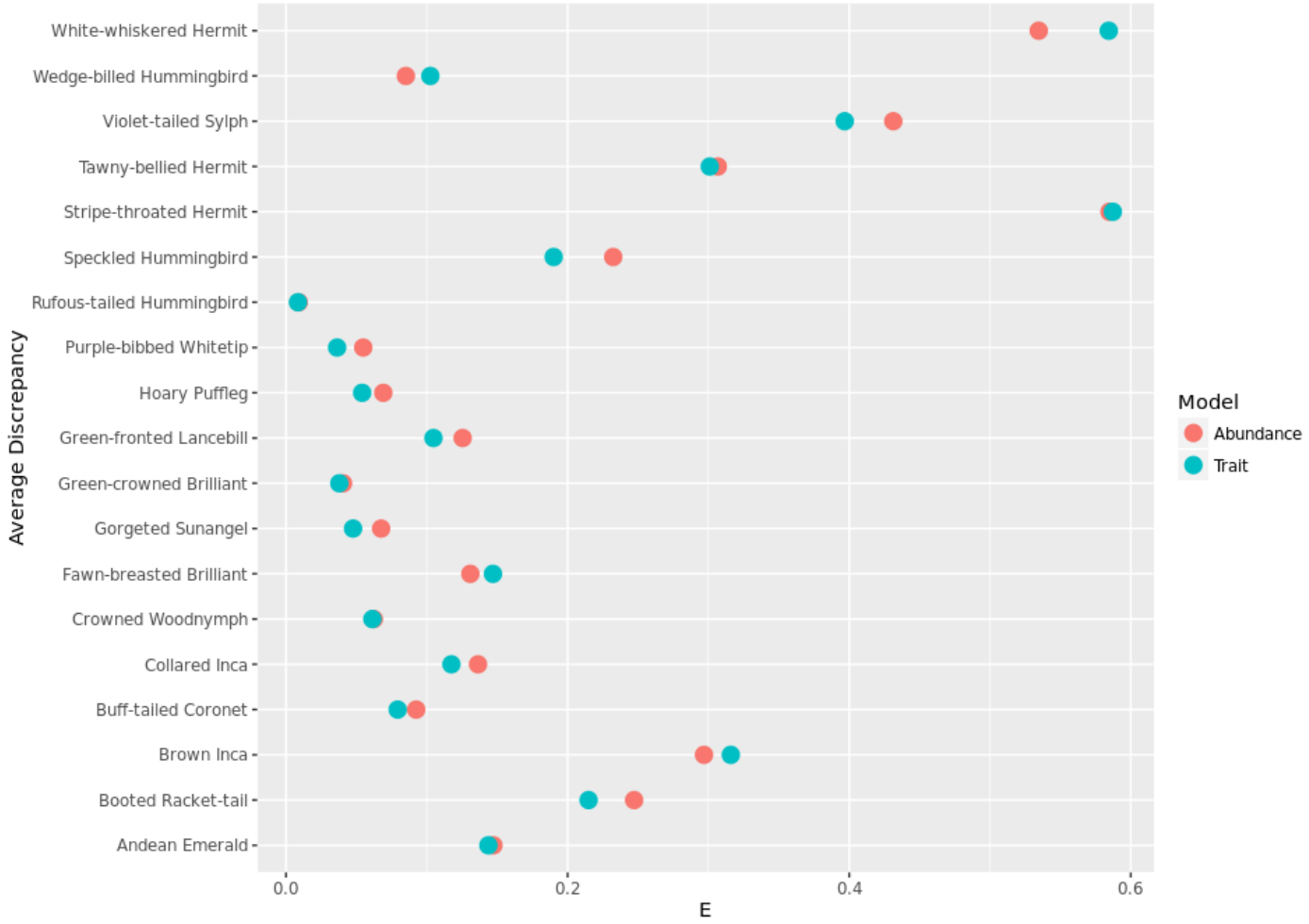

```
#By species
ynewstat<-merge(ynewstat,jagsIndexBird,by.x="species",by.y="jBird")
ynewstat<-merge(ynewstat,jagsIndexPlants,by.x="plant",by.y="jPlant")

ggplot(ynewstat,aes(x=trait,y=Abundance)) + geom_point() + geom_abline() + labs(x="Traits",y="Ab
undance") + ggtitle("Predicted Counts") + theme_bw() + facet_wrap(~Hummingbird,scales="free") +
coord_fixed()
```



8.1 Which species did we predict well?

8.1.1 By Bird

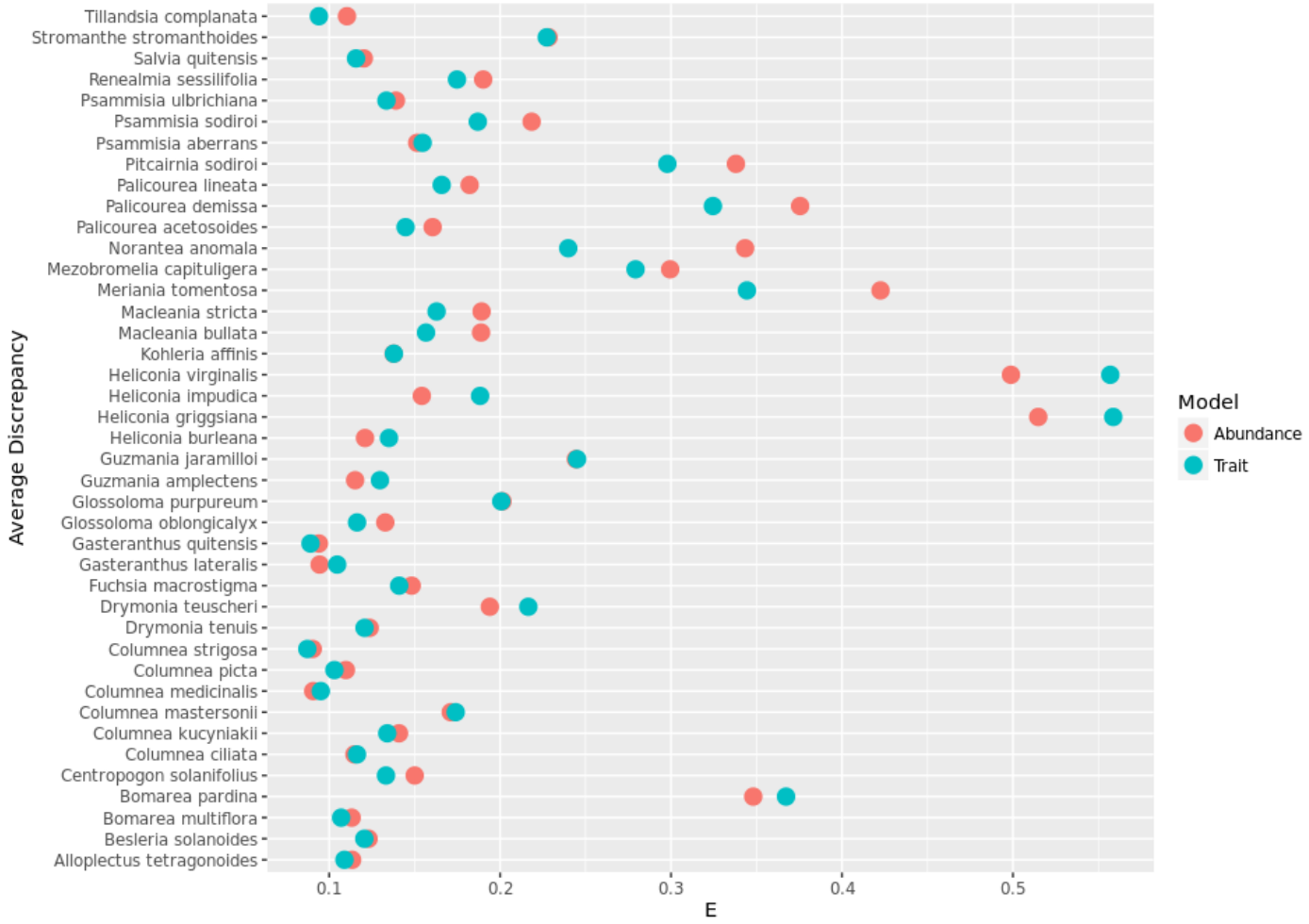
```
par_at<-list(Abundance=pars_abundance %>% filter(par=="E"),Trait=pars_detect_traits %>% filter(p
ar=="E"))
par_at<-melt(par_at,id.vars=colnames(pars_detect_traits))
dmat<-par_at %>% group_by(species,L1) %>% summarize(E=mean(estimate))
dmat<-merge(dmat,jagsIndexBird,by.x="species",by.y="jBird")

ggplot(dmat,aes(x=Hummingbird,y=E,col=L1)) + geom_point(size=4) + coord_flip() + labs(color="Mod
el",x="Average Discrepancy")
```




```
ggsave("Figures/AverageBird.jpeg",dpi=600,height=7,width=6)
```

8.1.2 By plant

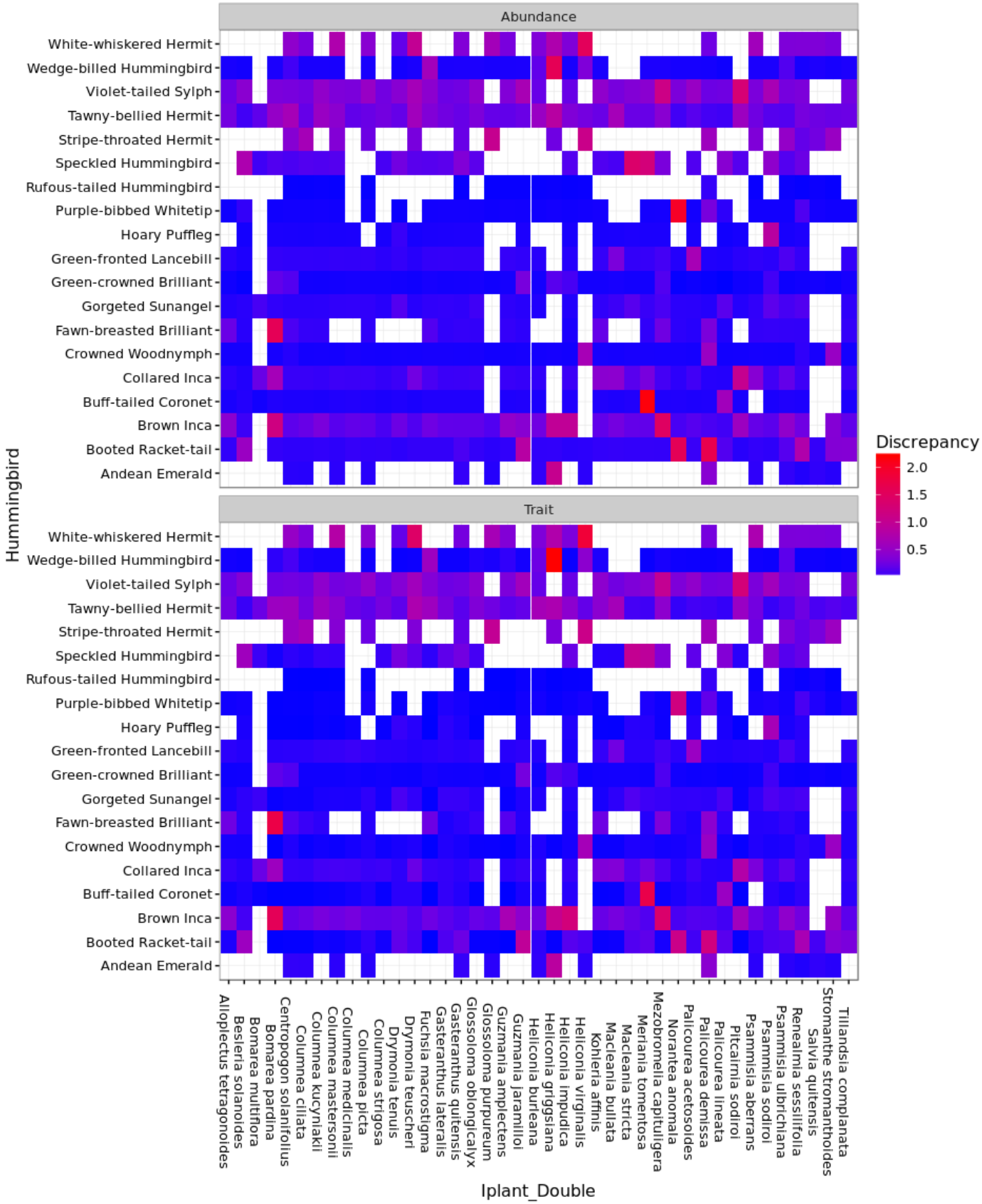
```
dmat<-par_at %>% group_by(plant,L1) %>% summarize(E=mean(estimate))
dmat<-merge(dmat,jagsIndexPlants,by.x="plant",by.y="jPlant")
ggplot(dmat,aes(x=Iplant_Double,y=E,col=L1)) + geom_point(size=4) + coord_flip()+ labs(color="Mo
del",x="Average Discrepancy")
```




```
ggsave("Figures/AveragePlant.jpeg",dpi=600,height=9,width=6)
```

8.1.3 By Interaction

```
dmat<-par_at %>% group_by(species,plant,L1) %>% summarize(E=mean(estimate))
dmat<-merge(dmat,jagsIndexPlants,by.x="plant",by.y="jPlant")
dmat<-merge(dmat,jagsIndexBird,by.x="species",by.y="jBird")
ggplot(dmat,aes(x=Iplant_Double,y=Hummingbird,fill=E)) + geom_tile(size=4) + scale_fill_continuo
us("Discrepancy",low='blue',high='red') + facet_wrap(~L1,nrow=2) + theme_bw() + theme(axis.text.
x=element_text(angle=-90))
```



8.1.4 Worst twenty fits.

```
print("Worst 20")
```

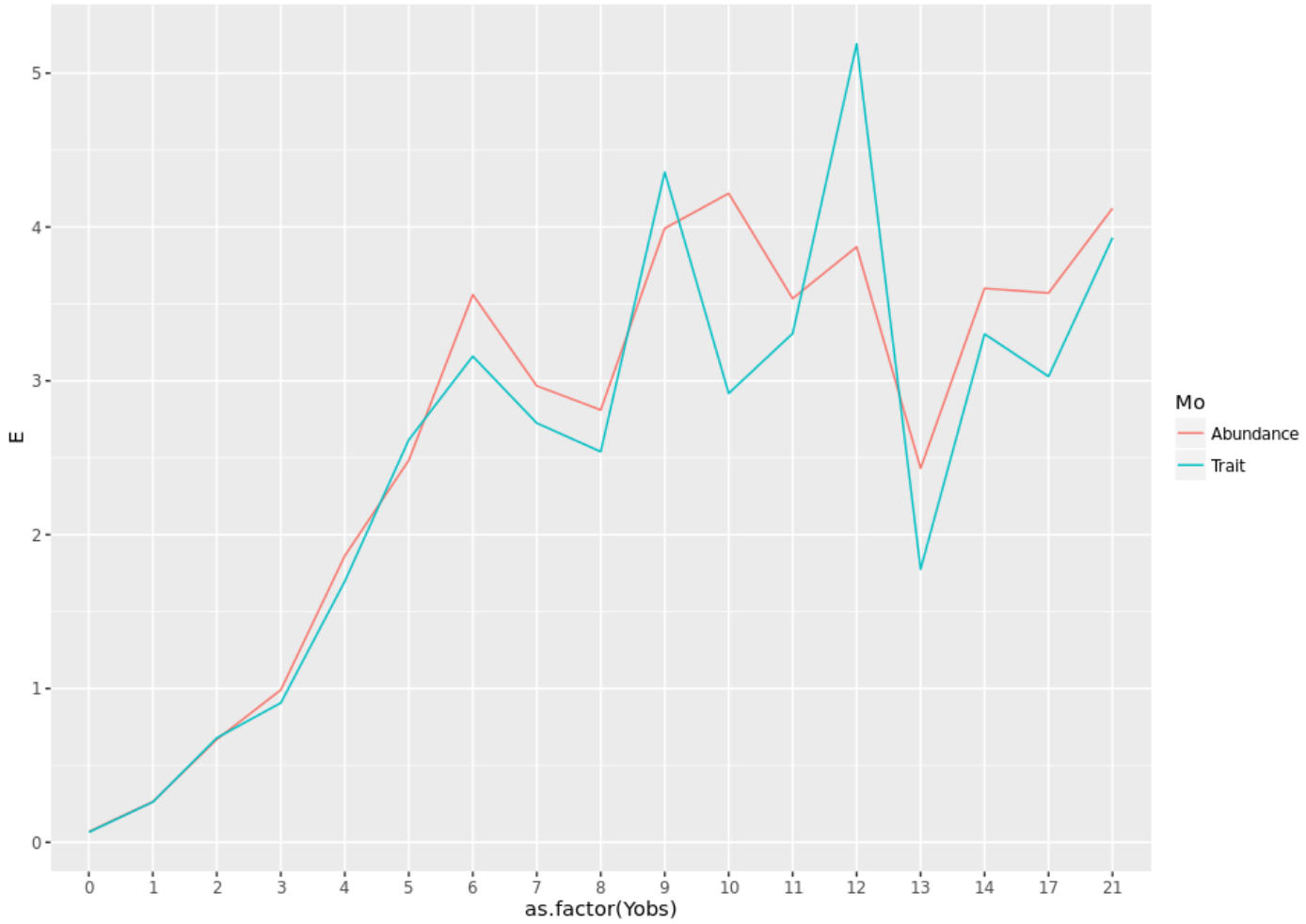
```
## [1] "Worst 20"
```

```
head(dmat %>% arrange(desc(E),L1),20)
```

```
##    species plant        L1        E             Iplant_Double
## 1       18    22     Trait 2.202544      Heliconia griggsiana
## 2        4    28 Abundance 2.165750        Meriania tomentosa
## 3       12    30 Abundance 2.003808          Norantea anomala
## 4       19    24     Trait 1.866510      Heliconia virginalis
## 5        7     4     Trait 1.736597           Bomarea pardina
## 6        4    28     Trait 1.707924        Meriania tomentosa
## 7        2    32 Abundance 1.673497        Palicourea demissa
## 8       18    22 Abundance 1.605085      Heliconia griggsiana
## 9        7     4 Abundance 1.596219           Bomarea pardina
## 10       2    30 Abundance 1.585374          Norantea anomala
## 11       3     4     Trait 1.582965           Bomarea pardina
## 12      19    24 Abundance 1.555214      Heliconia virginalis
## 13       3    29 Abundance 1.482977 Mezobromelia capituligera
## 14      19    13     Trait 1.434850        Drymonia teuscheri
## 15      14    27 Abundance 1.394203         Macleania stricta
## 16       3    29     Trait 1.387999 Mezobromelia capituligera
## 17      17    34 Abundance 1.303876        Pitcairnia sodiroi
## 18      17    34     Trait 1.266689        Pitcairnia sodiroi
## 19      14    28 Abundance 1.256513        Meriania tomentosa
## 20       3    23     Trait 1.246382        Heliconia impudica
##                Hummingbird
## 1  Wedge-billed Hummingbird
## 2       Buff-tailed Coronet
## 3    Purple-bibbed Whitetip
## 4    White-whiskered Hermit
## 5   Fawn-breasted Brilliant
## 6       Buff-tailed Coronet
## 7        Booted Racket-tail
## 8  Wedge-billed Hummingbird
## 9   Fawn-breasted Brilliant
## 10       Booted Racket-tail
## 11               Brown Inca
## 12   White-whiskered Hermit
## 13               Brown Inca
## 14   White-whiskered Hermit
## 15     Speckled Hummingbird
## 16               Brown Inca
## 17      Violet-tailed Sylph
## 18      Violet-tailed Sylph
## 19     Speckled Hummingbird
## 20               Brown Inca
```

8.1.5 By Intensity

```
dmat<-par_at %>% group_by(Yobs,L1) %>% summarize(E=mean(estimate))
ggplot(data=dmat,aes(x=as.factor(Yobs),y=E)) + geom_line(aes(col=L1,group=L1)) + labs("Observed",c
l="Mo")
```



8.2 Detection table

```
dp<-group_by(pars_detect_traits[pars_detect_traits$par %in% c("detect"),],species) %>% summarise(m
an=round(mean(estimate,na.rm=T),3)*100,lower=round(quantile(estimate,0.025,na.rm=T),3)*100,upper=r
ound(quantile(estimate,0.975,na.rm=T),3)*100)

tab<-merge(dp,jagsIndexBird,by.x="species",by.y="jBird")[,-1]
tab[,c(4,1,2,3)]
```

```
##                  Hummingbird mean lower upper
## 1              Andean Emerald 42.3  14.2  72.9
## 2          Booted Racket-tail 24.0   9.6  43.7
## 3                  Brown Inca 35.7  22.0  52.6
## 4         Buff-tailed Coronet 36.1  16.0  59.8
## 5               Collared Inca 50.0  25.4  73.4
## 6           Crowned Woodnymph 27.8   9.8  46.7
## 7    Fawn-breasted Brilliant 30.4   9.6  63.9
## 8           Gorgeted Sunangel 70.0  47.4  87.1
## 9    Green-crowned Brilliant 28.1   5.9  60.7
## 10   Green-fronted Lancebill 44.7  23.7  65.7
## 11              Hoary Puffleg 36.4  14.6  68.1
## 12    Purple-bibbed Whitetip 38.4  14.1  68.4
## 13 Rufous-tailed Hummingbird 28.7   2.9  66.5
## 14      Speckled Hummingbird 50.0  14.4  87.7
## 15    Stripe-throated Hermit 26.1  13.6  42.0
## 16      Tawny-bellied Hermit 33.8  22.5  50.6
## 17       Violet-tailed Sylph 40.3  29.4  54.8
## 18  Wedge-billed Hummingbird 17.7   3.1  47.1
## 19    White-whiskered Hermit 21.7   8.8  35.9
```

```
write.csv(tab[,c(4,1,2,3)],"Figures/Table1.csv")
```

8.3 Sampling intensity and detection for each hummingbird species

The probability of missing a species at each daily visit is 1 - detection probability.

The probability of missing a species on sequential visits is (1- detection probability) * (1 - detection probability).

We are interested in the number of sampling events that minimize this value to some reasonable threshold. I have chosen 0.05 by convention.

The following figure represesent the estimated number of daily surveys to capture a hummingbird event given that we know it occurs. These data can be thought of as successful draws from a negative binomial distribution.

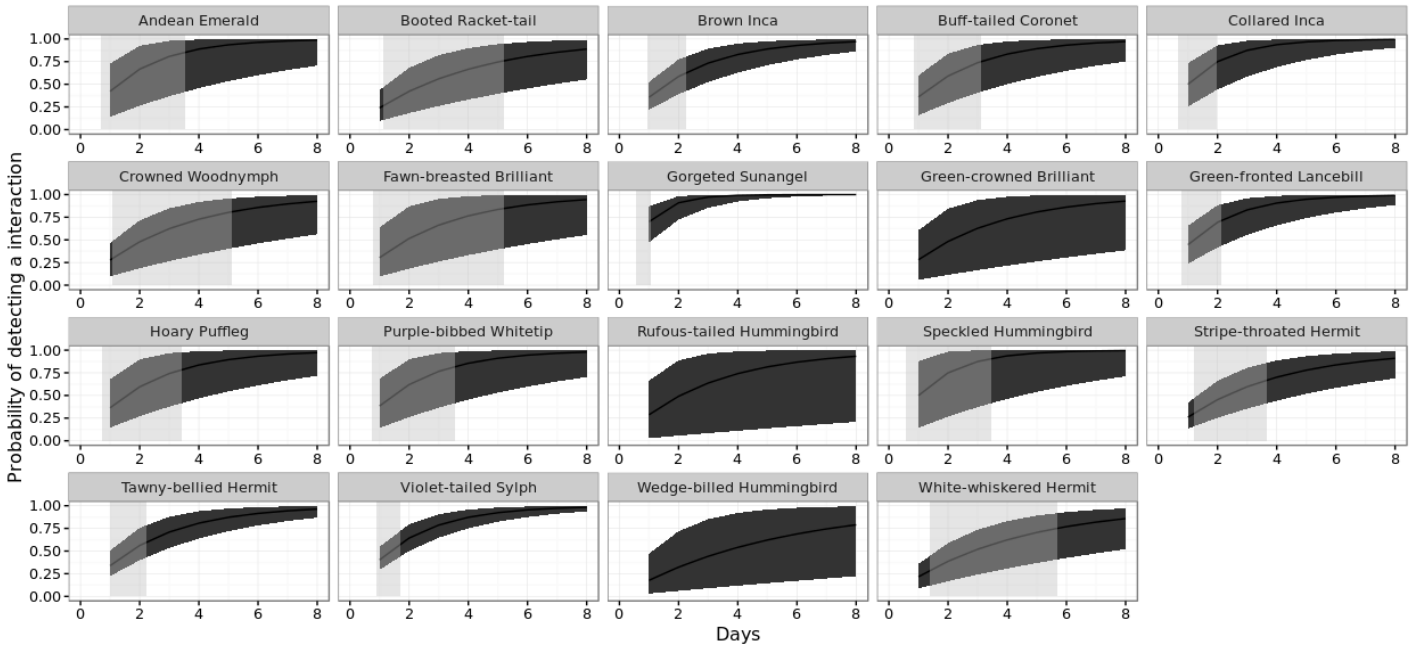
It is easiest to interpret as the number of days until you are likely to see an interaction, so i prefer to calculate:

$$1-(1-p)^n$$

$$p = probabilityofdetection$$

$$n = dayssample$$

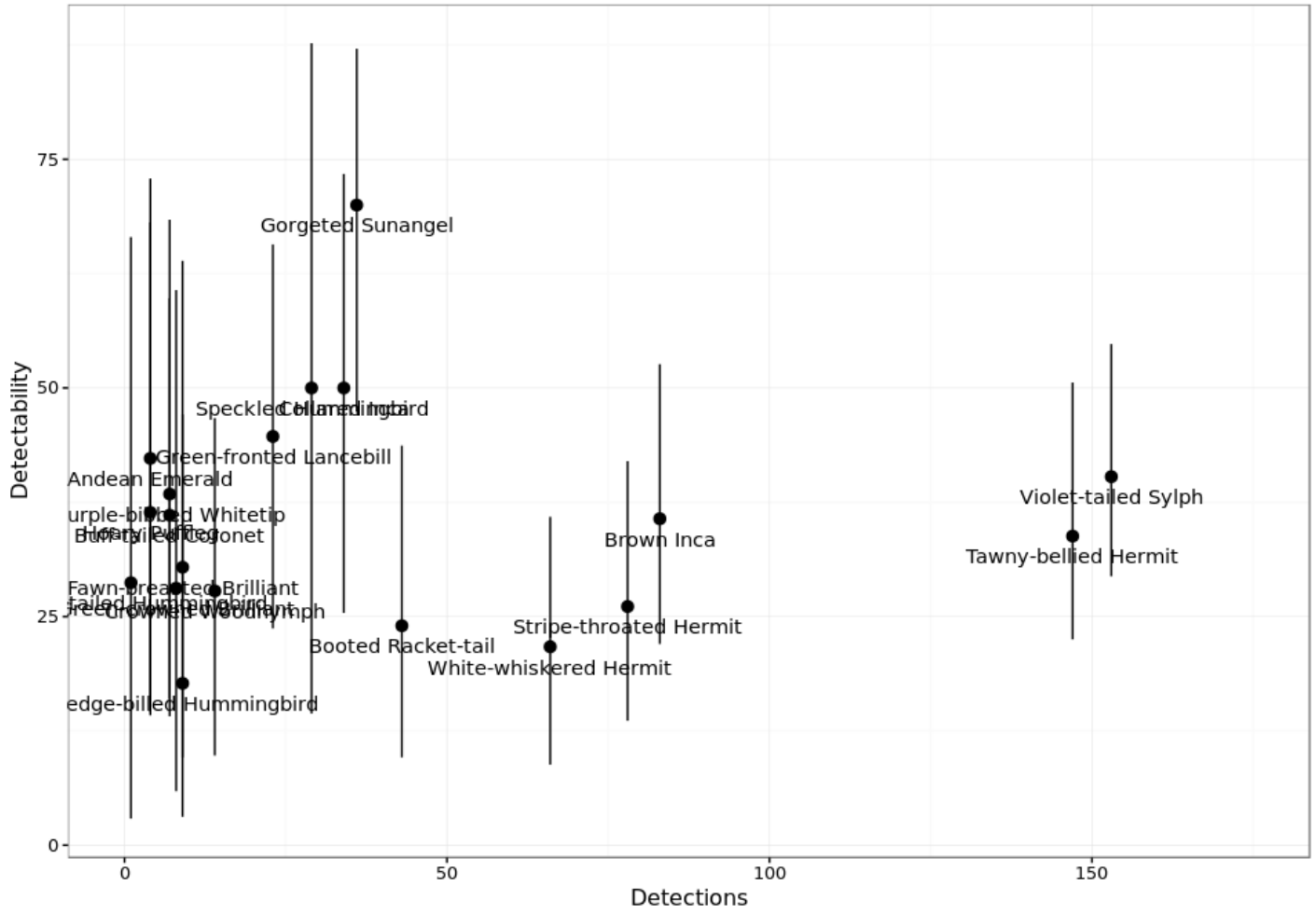
```
dp<-function(n,p){
  1-((1-p)^n)
}

ts<-split(tab,tab$Hummingbird,drop=T)
detectd<-lapply(ts,function(x){
  meanD<-dp(n=1:10,p=x$mean/100)
  lowerD<-dp(n=1:10,p=x$lower/100)
  upperD<- dp(n=1:10,p=x$upper/100)
  data.frame(Days=1:10,mean=meanD,lower=lowerD,upper=upperD)
})

md<-melt(detectd,id.var="Days")
md<-dcast(md,...~variable)

#get the 0.5 line
dpn<-function(t,p){
  n<-(1 - (1-t))/(p/100)
  return(n)
}

#for each bird get the upper and middle estimate for 50% chance.
daydf<-list()
for (x in 1:nrow(tab)){
  mean_day=dpn(t=0.5,tab$mean[x])
  lower_day=dpn(t=0.5,tab$lower[x])
  upper_day=dpn(t=0.5,tab$upper[x])
  daydf[[x]]<-data.frame(L1=tab$Hummingbird[x],mean=mean_day,lower=lower_day,upper=upper_day)
}
daydf<-rbind_all(daydf)

ggplot(md) + geom_ribbon(aes(x=Days,y=mean,ymin=lower,ymax=upper)) + geom_line(aes(x=Days,fill=L
1,y=mean,ymin=lower,ymax=upper)) + facet_wrap(~L1,nrow=4,scale="free_x")  + ylab("Probability of
 detecting a interaction") + scale_fill_discrete(guide="none") + theme_bw() + scale_x_continuous(b
eaks=seq(0,8,2),limits=c(0,8))+ geom_rect(fill='grey',data=daydf,alpha=0.4,aes(xmax=upper,xmin=l
ower,ymin=0,ymax=Inf)) + ylim(0,1)
```




```
ggsave("Figures/DetectionDays.jpeg",height=7,width=11,dpi=300)
```

The number of days it would take to have 50% confidence you have sampled enough to capture known interactions is the x axis value where the dotted line hits the curve.

```
sampling<-indatraw %>% group_by(Hummingbird) %>% summarize(Obs=length(Hummingbird))

tabD<-merge(tab,sampling,by="Hummingbird")
ggplot(tabD,aes(x=Obs,ymin=lower,ymax=upper,y=mean)) + geom_pointrange() + labs(y="Detectabilit
y",x="Detections") + geom_text(aes(label=Hummingbird),vjust=2) + theme_bw() + xlim(0,175)
```



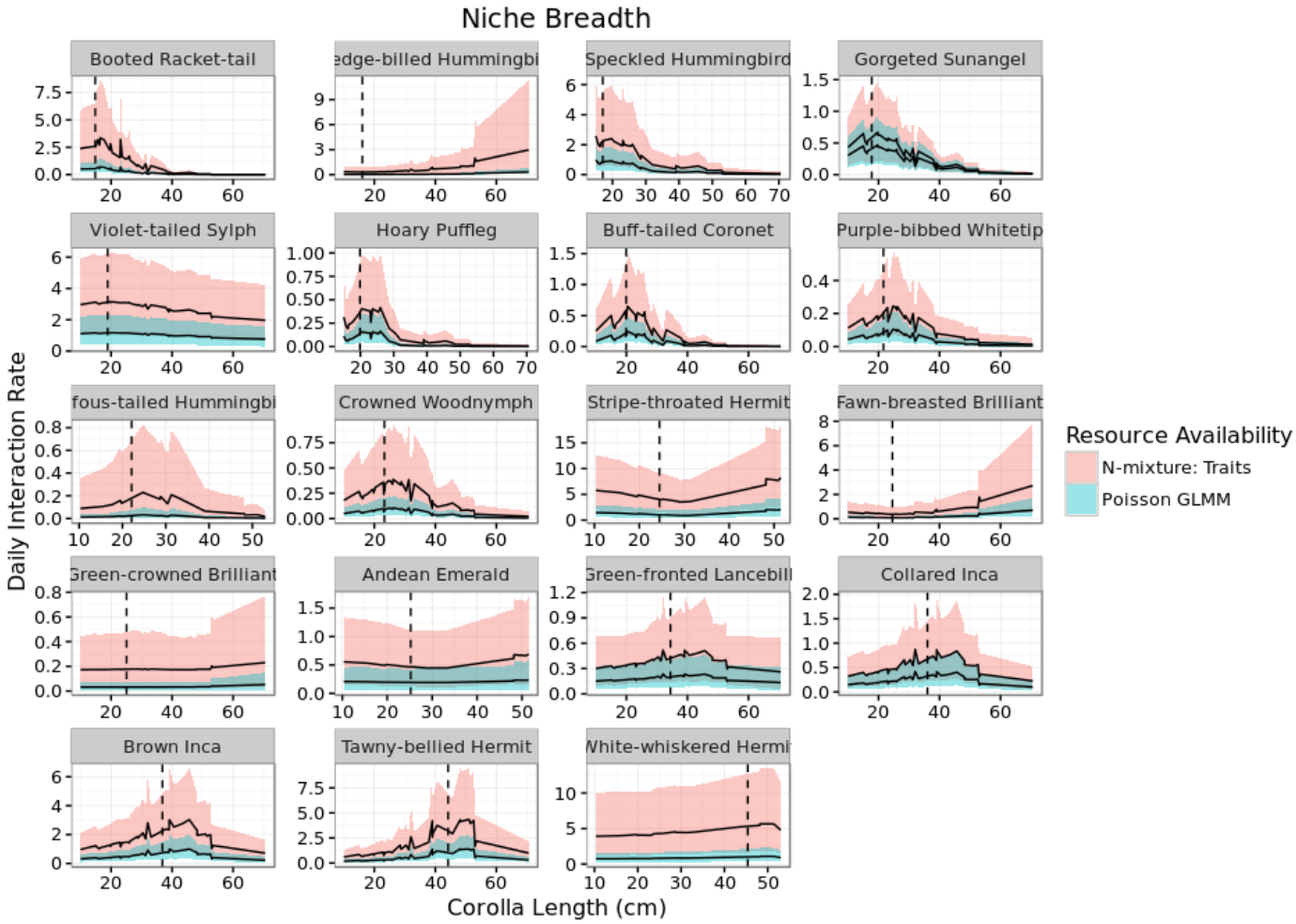
9 Predicted versus Observed Data

```
m<-max(jTraitmatch)-jTraitmatch

mat<-indat %>% group_by(jBird,jPlant) %>% summarize(n=sum(Yobs))
true_state<-acast(mat,jBird~jPlant,fill=0)
```

9.1 Generate Networks

```
castdf<-dcast(parsObs[parsObs$par %in% c("beta1","alpha"),], species +Model+Chain + Draw~par,val
ue.var="estimate")

tcastdf<-dcast(parsObs[parsObs$par %in% c("tauE"),],  Chain +Model+ Draw~par,value.var="estimat
e")

castdf<-merge(castdf,tcastdf,by=c("Chain","Model","Draw"))

#Turn to
castdf$species<-factor(castdf$species,levels=1:max(as.numeric(castdf$species)))

species.split<-split(castdf,list(castdf$species,castdf$Model),drop = T)

species.traj<-lapply(species.split,function(dat){
  index<-unique(dat$species)

  #get data for those species
  billd<-indat[indat$jBird %in% index,]

  d<-data.frame(alpha=dat$alpha,beta1=dat$beta1,tauSigma=1/sqrt(dat$tauE))

  #fit regression for each input estimate
  sampletraj<-list()

  for (y in 1:nrow(d)){
    v=exp(d$alpha[y] + d$beta1[y] * billd$Traitmatch + rnorm(1,0,d$tauSigma))

    sampletraj[[y]]<-data.frame(x=as.numeric(billd$Traitmatch),y=as.numeric(v),jBird=billd$jBird,j
lant=billd$jPlant,Model=unique(dat$Model))
  }

  sample_all<-rbind_all(sampletraj)
})

species.traj<-rbind_all(species.traj)

species.mean<-species.traj %>% group_by(jBird,jPlant,Model) %>% summarize(Traitmatch=unique(x),p
hi=mean(y))

species.mean<-merge(species.mean,indat[,colnames(indat) %in% c("jBird","jPlant","jTime","Humming
bird","Iplant_Double")])

#get corolla sizes
species.mean<-merge(species.mean,fl.morph,by.x="Iplant_Double", by.y="Group.1")

#bill order
ord<-hum.morph %>% arrange(Total_Culmen) %>% .$English
species.mean$Hummingbird<-factor(species.mean$Hummingbird,levels=ord)

#add level to hum.morph to match naming convention
species.mean<-merge(species.mean,hum.morph[,c("English","Total_Culmen")],by.x="Hummingbird",by.y="
nglish")
```

```
#Niche Breadth
species.mean<-species.traj %>% group_by(jBird,jPlant,Model) %>% summarize(Traitmatch=unique(x),p
hi=mean(y),phi_low=quantile(y,0.05),phi_high=quantile(y,0.95))

#merge names
species.mean<-merge(species.mean,jagsIndexBird)
species.mean<-merge(species.mean,jagsIndexPlants)

#get corolla sizes
species.mean<-merge(species.mean,fl.morph,by.x="Iplant_Double", by.y="Group.1")

#bill order
ord<-hum.morph %>% arrange(Total_Culmen) %>% .$English
species.mean$Hummingbird<-factor(species.mean$Hummingbird,levels=ord)

#add level to hum.morph to match naming convention
species.mean<-merge(species.mean,hum.morph[,c("English","Total_Culmen")],by.x="Hummingbird",by.y="
nglish")

ggplot(species.mean) +
geom_ribbon(alpha=0.4,aes(x=TotalCorolla,ymin=phi_low,ymax=phi_high,fill=as.factor(Model))) + th
eme_bw() + facet_wrap(~Hummingbird,scales="free",ncol=4)+ ggtitle("Niche Breadth") +
geom_vline(aes(xintercept=Total_Culmen),linetype='dashed') +
geom_line(aes(x=TotalCorolla,y=phi,fill=as.factor(Model))) + ylab("Daily Interaction Rate") + xl
ab("Corolla Length (cm)") + scale_fill_discrete("Resource Availability")
```

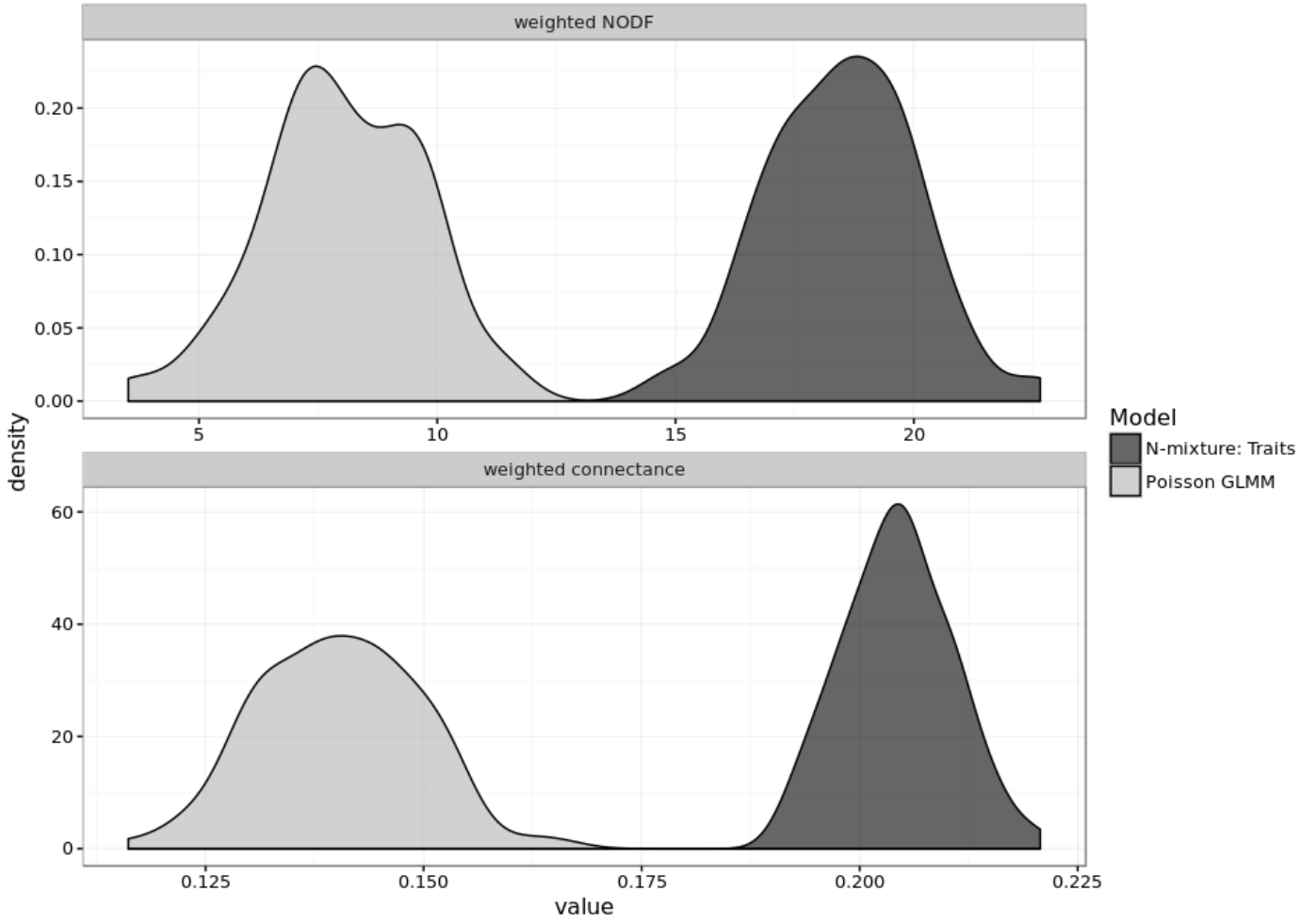



```
ggsave("Figures/NicheBreadth.jpeg",height=6,width=9)
```

10 Network Statistics

Given the uncertainty in species interactions, what do emergant statistics look like?

```
#Split by resource
nsplit<-split(species.mean,species.mean$Model)

makeN<-function(x){

  #input matrix
  aggm<-matrix(nrow=nrow(jagsIndexBird),ncol=nrow(jagsIndexPlants),data=0)
  for (j in 1:nrow(x)){
    aggm[x[j,"jBird"],x[j,"jPlant"]]<-rpois(1,lambda=x[j,"phi"])
  }
  #calculate network statistic
  nstat<-networklevel(aggm,index=c("weighted connectance","weighted NODF"))
}

nstat<-lapply(nsplit,function(x){
  netstat<-melt(t(sapply(1:100,function(k) makeN(x))))
  colnames(netstat)<-c("Iteration","Metric","value")
  return(netstat)
})

nstat<-melt(nstat,colnames(nstat[[1]]))

ggplot(nstat,aes(x=value,fill=L1)) + geom_density(alpha=0.6) +
facet_wrap(~Metric,scales='free',nrow=2)  +
scale_fill_manual("Model",values=c("black","grey70")) + theme_bw()
```



```
ggsave("Figures/NetworkStats.jpeg",height = 5,width=6)
```

```
gc()
```

```
##             used   (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells   1785379   95.4    5947311  317.7  13178414  703.9
## Vcells 215329709 1642.9  347018806 2647.6 347018805 2647.6
```

```
save.image("Observed.RData")
```